RECURSIVE CONSTRAINT COMPILATION FOR GUIDED PROTEIN BACKBONE DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Many protein design problems require designing a flexible backbone region while leaving the rest of the protein structurally and functionally intact. This is especially important in tasks such as antibody-loop design, fragment inpainting, and motif-linker design, where a useful design must also connect exactly to fixed structural boundaries. Existing methods for constrained generation can enforce hard geometric constraints but typically do so by modifying the generative process itself or by repairing the structures they have generated. To address this gap, we introduce Recursive Constraint Compilation (RCC), a generator-independent framework that recursively decomposes a global hard constraint into local compatibility conditions and assembles their parameterizations into coordinates for valid structures. For fixed-endpoint protein backbones, RCC provides exact, invertible coordinates on regular regions of the constrained space, enabling probability modeling, generation, and guidance while maintaining the geometric constraints throughout inference. We evaluate RCC across pretrained protein generators for antibody-loop design, fragment inpainting, and motif-linker design, as well as in prospective CDR-H3 design. Across these settings, RCC preserves the behavior of existing generators while enforcing exact geometry and enables effective guidance over valid backbones that is not recovered by unconstrained Cartesian optimization followed by *post-hoc* repair.

1 INTRODUCTION

Many protein-design problems require redesigning a local backbone segment while preserving the surrounding structure and function of the protein. This is central to tasks such as antibody-loop design, fragment inpainting, and motif-linker design, where the model is free to redesign one region but the new backbone must still connect exactly to the fixed structure around it (Luo et al., 2022; Wang et al., 2022; Watson et al., 2023). Although generative backbone models can produce diverse, structurally plausible backbones (Boom et al., 2023; Yim et al., 2023a;b), local redesign also requires exact endpoint and link-length constraints. This matters in guided design because an optimizer can improve its objective by exploiting geometries that violate the boundary conditions. Enforcing those conditions afterward can then disrupt the structural features responsible for the improvement.

Constrained generative methods handle these requirements in several ways: they define generative dynamics directly in the constrained space, map unconstrained variables into that space, or repair samples during or after generation (De Bortoli et al., 2022; Huang et al., 2022; Chen & Lipman, 2024; Feng et al., 2025; Li et al., 2026; Christopher et al., 2026). These approaches can produce valid outputs but are often tied to a particular generative procedure, constraint-specific transformation, or correction step. Our goal is to develop a reusable constraint representation that can be applied to pretrained protein generators while supporting generation, probability modeling, and guidance entirely within the valid space.

We introduce **Recursive Constraint Compilation (RCC)**, a generator-independent framework that recursively decomposes a global hard constraint into local compatibility conditions and composes their solutions into explicit coordinates for valid structures. For protein backbones, RCC recursively partitions the chain into shorter segments. Each segment is summarized by its endpoint displacement, and local coordinates specify a compatible split into two child segments. Recursing over these splits yields exact, invertible coordinates for the complete fixed-endpoint backbone on regular domains. The

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.

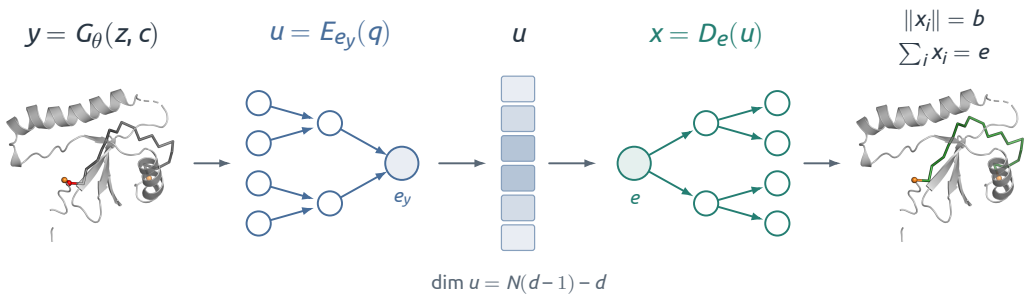


Figure 1: Overview of RCC for fixed-endpoint protein backbones. Generated links y are normalized to a source bridge q with endpoint displacement e_y. RCC encodes q as u and decodes the same coordinates under the required displacement e. The resulting backbone has the prescribed endpoints and link lengths, while $\mathcal{J}_e$ provides the volume correction relating coordinate and backbone densities.

same recursion also factors the intrinsic volume of the constrained space, so the resulting coordinates support normalized density evaluation, sampling, and differentiable guidance in this representation (Figure 1). Because the construction is independent of the learned model, RCC can be applied to pretrained generators without retraining while satisfying hard geometric constraints throughout inference.

We summarize our main contributions as follows.

1. **Recursive constraint compilation.** We introduce a generator-independent framework that recursively assembles local constraint parameterizations into exact, invertible coordinates on regular domains. We establish conditions for this construction and apply it to protein backbones and bounded fixed-total allocations.
2. **Probabilistic coordinates for protein backbones.** We derive explicit backbone encoders and decoders and a recursive intrinsic-volume factor. The resulting coordinates support normalized density evaluation, sampling, and differentiable guidance under exact endpoint and link-length constraints.
3. **Guided protein backbone design.** We apply RCC to pretrained generators for antibody loops, fragment inpainting, and motif linkers without retraining, with structural measures close to those obtained by fixed projection. In prospective CDR-H3 experiments, feasible guidance improves structural success over learned sampling. Matched *post-hoc* comparisons show that Cartesian optimization followed by repair does not recover these gains.

2 Recursive Constraint Compilation

RCC converts a hard global constraint into a hierarchy of smaller local constraints. It recursively divides a configuration into pieces, records how those pieces fit together, and reconstructs them so that the full constraint is satisfied. The resulting coordinates describe only feasible configurations and can be used during both generation and optimization. We first establish when this construction is exact and invertible and then apply it to protein backbones with fixed endpoints.

2.1 Recursive compilation of hard constraints

Consider a hard constraint $F : \mathcal{X} \to \mathcal{Y}$ with target s, defining

$$\mathcal{M}_s = \{x \in \mathcal{X} : F(x) = s\}. \tag{1}$$

RCC constructs coordinates for $\mathcal{M}_s$ by recursively dividing a configuration into smaller pieces. We represent this decomposition by a rooted binary tree. Each node v contains a subconfiguration x_v and is assigned a summary

$$H_v(x_v) \in \mathcal{S}_v, \tag{2}$$

with $\dim \mathcal{S}_v \le k$, independent of the size of the subtree. At the root, we take $\mathcal{S}_{\text{root}} = \mathcal{Y}$ and $H_{\text{root}} = F$, so setting the root summary to s enforces the original constraint. For every other node, the summary records only what is needed to combine that subtree with the rest of the configuration.

Suppose an internal node v combines children ℓ and r. Their summaries must determine the summary of the parent:

$$H_v(x_\ell \star x_r) = m_v(H_\ell(x_\ell), H_r(x_r))\,, \tag{3}$$

where $\star$ joins the two child configurations. If the parent must have summary $s_v \in \mathcal{S}_v$, the allowed pairs of child summaries are

$$\mathcal{G}_v(s_v) = \{(s_\ell, s_r) \in \mathcal{S}_\ell \times \mathcal{S}_r : m_v(s_\ell, s_r) = s_v\}\,. \tag{4}$$

RCC describes these allowed choices using local coordinates. On a region where the parameterization is well defined and invertible, we write

$$g_{v,s_v} : \mathcal{Z}_v \to \mathcal{G}_v^{\text{reg}}(s_v), \qquad a_{v,s_v} : \mathcal{G}_v^{\text{reg}}(s_v) \to \mathcal{Z}_v \tag{5}$$

for the decoder and encoder at node v. The coordinate domain $\mathcal{Z}_v$ is the same for different values of the parent summary s_v.

Definition 2.1 (Recursive compilability)**.** A constraint family is recursively compilable on a regular domain if each configuration can be divided into a tree whose subtrees have bounded-dimensional summaries, and each parent summary is determined by the summaries of its two children. At every internal node, the compatible child summaries must admit invertible local coordinates from a domain that does not depend on the parent summary. Each leaf must also admit exact, invertible coordinates for its assigned summary.

Given a target s and a set of local coordinates, the decoder starts at the root and uses g_{v,s_v} to choose compatible summaries for the two children at each split. Repeating this process determines the summaries at the leaves, where the leaf decoders construct the final pieces. The encoder follows the same tree in reverse.

Theorem 2.2 (Recursive chart assembly)**.** *Assume the conditions of Definition 2.1. On a balanced tree with n leaves, the local maps assemble into global encoder and decoder maps that are mutually inverse on their regular domains, and every decoded configuration satisfies $F(x) = s$. If each node uses $O(k)$ work, encoding and decoding require $O(nk)$ total work and $O(k \log n)$ parallel depth.*

The constraint is satisfied because the child summaries at every split recombine to the summary assigned to their parent, ending with $H_{\text{root}} = F = s$. The inverse follows by recovering the same local choices in reverse. The construction therefore enforces the global constraint without solving a new global feasibility problem at each decode. Appendix A.1 gives the proof and the conditions under which the local maps are differentiable.

2.2 Coordinates for fixed-endpoint protein backbones

In local backbone design, the surrounding structure fixes the two ends of the segment. The new backbone must connect these endpoints while maintaining the prescribed Cα spacing. RCC handles this constraint by recursively splitting the segment into shorter chains and choosing where they meet.

Let $p_0, \dots, p_N \in \mathbb{R}^d$ denote the Cα positions, with link vectors $x_i = p_i - p_{i-1}$ and endpoint displacement $e = p_N - p_0$. For a prescribed link length $b > 0$, the feasible backbones are

$$\mathcal{B}_N(e) = \left\{ (x_1, \dots, x_N) \in (\mathbb{S}_b^{d-1})^N : \sum_{i=1}^{N} x_i = e \right\}, \tag{6}$$

where $\mathbb{S}_b^{d-1}$ is the sphere of radius b. We assume $N \ge 2$, $d \ge 2$, and $\|e\| < Nb$. At noncollinear configurations, $\mathcal{B}_N(e)$ has dimension $N(d-1) - d$.

For this constraint, a subchain is summarized by its endpoint displacement. Consider an n-link chain from 0 to D, split into $n_\ell \ge 2$ left links and $n_r \ge 2$ right links, with $n = n_\ell + n_r$. If the two subchains meet at X, their displacements are X and $D - X$. The left subchain can reach X only if $\|X\| \le n_\ell b$, and the right can connect X to D only if $\|D - X\| \le n_r b$. The possible meeting points are therefore

$$\mathcal{L}(D; n_\ell, n_r) = \overline{B}(0, n_\ell b) \cap \overline{B}(D, n_r b). \tag{7}$$

Once X is chosen, the original chain separates into two shorter fixed-endpoint problems.

We now give these meeting points a coordinate system with the same domain for different endpoint displacements. Set

$$c = \frac{n_\ell}{n} D. \tag{8}$$

Because $\|D\| < nb$, c lies inside both balls. For a unit direction $u \in \mathbb{S}^{d-1}$, let $\rho_D(u)$ be the distance from c to the boundary of $\mathcal{L}(D; n_\ell, n_r)$ along u. A radial coordinate $r \in (0, 1)$ then determines

$$X = c + r\rho_D(u)u. \tag{9}$$

The coordinates (u, r) have the same domain for every D, while $r < 1$ ensures that X lies inside the feasible region. For $X \neq c$, they are recovered by

$$u = \frac{X - c}{\|X - c\|}, \tag{10}$$

$$r = \frac{\|X - c\|}{\rho_D(u)}. \tag{11}$$

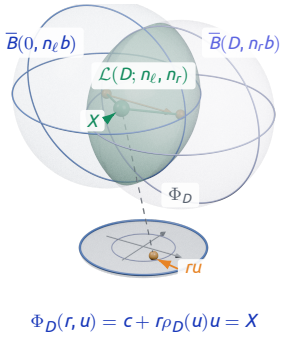


Figure 2: Meeting-point coordinates for a fixed-endpoint backbone. The overlap of the two balls gives the possible meeting points. A direction u and radial coordinate r determine X within this overlap.

Applying the same construction to the two child chains reduces the problem to two- and three-link chains.

The split coordinates and the coordinates of the final two- and three-link chains together have dimension $N(d-1) - d$, matching the dimension of $\mathcal{B}_N(e)$. Let $\mathcal{U}_N$ denote their coordinate domain and $\mathcal{U}_N^{\mathrm{reg}}(e) \subseteq \mathcal{U}_N$ the subset where the recursive maps are well defined and invertible. For endpoint displacement e, the decoder

$$D_e : \mathcal{U}_N^{\mathrm{reg}}(e) \to \mathcal{B}_N(e) \tag{12}$$

constructs a valid backbone, and the encoder E_e recovers its coordinates.

Proposition 2.3 (Exact bridge coordinates)**.** *For every $N \geq 2$, $d \geq 2$, $b > 0$, and $\|e\| < Nb$,*

$$D_e(u) \in \mathcal{B}_N(e), \qquad D_e(E_e(x)) = x, \qquad E_e(D_e(u)) = u \tag{13}$$

whenever the corresponding coordinates are well defined.

Appendix A.2 gives the two- and three-link constructions and the corresponding regularity conditions. The results and experiments in the main text use equal link lengths.

2.3 Enforcing constraints for pretrained protein generators

A pretrained generator can produce a backbone with incorrect Cα spacing or endpoint displacement. Let $y = G_\theta(z, c)$ be a generated sample with link vectors $y_1, \ldots, y_N$, and let e be the required endpoint displacement. RCC first converts y into an equal-link chain that can be encoded. We distribute the endpoint mismatch across the links and then restore each link to length b:

$$\delta = \frac{e - \sum_j y_j}{N}, \qquad q_i = b\frac{y_i + \delta}{\|y_i + \delta\|}, \qquad e_y = \sum_i q_i. \tag{14}$$

The resulting chain $q = (q_1, \ldots, q_N)$ has link length b and endpoint displacement e_y. RCC encodes q under e_y and decodes the same coordinates under the required displacement e:

$$R_e(y) = D_e\big(E_{e_y}(q)\big). \tag{15}$$

If $y_i + \delta \neq 0$ for every i and the encoder and decoder remain in compatible regular charts, then $R_e(y) \in \mathcal{B}_N(e)$. If the generated sample already belongs to $\mathcal{B}_N(e)$, then $\delta = 0$, $q = y$, and $e_y = e$, giving

$$R_e(y) = D_e(E_e(y)) = y. \tag{16}$$

Thus R_e is a retraction onto $\mathcal{B}_N(e)$ on its regular domain. Rather than minimizing Euclidean displacement from the generated sample, RCC keeps the recursive coordinates of q fixed while changing its endpoint displacement from e_y to e. This construction can be used with different

pretrained generators without retraining or changing their sampling procedures. Appendix A.2.5 gives the complete algorithm.

The retraction in Equation (15) acts after generation. For guided design, however, every candidate should satisfy the constraint when it is scored rather than only after optimization has finished. The same coordinates provide this guarantee and also support normalized density evaluation.

2.4 Probabilistic modeling and inference-time guidance

A density over RCC coordinates is not equivalent to a density over backbone shapes because the decoder can stretch different regions of coordinate space by different amounts. RCC computes this factor recursively from the same tree used to construct the chain, avoiding full Jacobian calculation of the backbone decoder.

At each split, the volume depends on variation within the two child subchains and on how both children deform when their meeting point moves. For a subtree I, write $\widehat{x}_i = x_i/b$ and define

$$M_I = \sum_{i \in I} \left(I_d - \widehat{x}_i \widehat{x}_i^\top \right). \tag{17}$$

Each term projects onto directions perpendicular to link i, along which that link can rotate without changing its length. The matrix M_I therefore describes how these link rotations can move the subchain endpoint. At an internal node v, let $B_v = \partial X / \partial \zeta_v$ describe how the local split coordinates ζ_v move the meeting point X. The parent volume factor then separates into the following terms.

Theorem 2.4 (Recursive bridge volume)**.** *On a one-to-one regular chart with noncollinear child configurations and a nonsingular meeting-point Jacobian B_v, the parent quantities satisfy*

$$\mathcal{J}_I = \underbrace{\mathcal{J}_\ell \mathcal{J}_r}_{\substack{\text{child volume} \\ \text{factors}}} \quad \underbrace{|\det B_v|}_{\substack{\text{meeting-point} \\ \text{volume scaling}}} \quad \underbrace{\sqrt{\det\left(M_\ell^{-1} + M_r^{-1}\right)}}_{\substack{\text{subchain} \\ \text{deformation}}}, \tag{18}$$

$$M_I = M_\ell + M_r. \tag{19}$$

The two- and three-link charts provide the base volume factors, and propagating them up a balanced tree requires $O(Nd^3)$ work and $O(d^3 \log N)$ parallel depth. For protein backbones, $d = 3$, giving linear work and logarithmic parallel depth.

Let $q_\psi(u \mid c, e)$ be a normalized coordinate density conditioned on protein context c and endpoint displacement e. For $u \sim q_\psi(\cdot \mid c, e)$ and $x = D_e(u)$, the corresponding density on the fixed-endpoint backbone space, measured with respect to intrinsic volume, is

$$q_{\mathcal{B}}(x \mid c, e) = \frac{q_\psi(E_e(x) \mid c, e)}{\mathcal{J}_e(E_e(x))}. \tag{20}$$

The same coordinates can be used for guidance without leaving the feasible space. Suppose $u = T_\psi(\varepsilon; c, e) \in \mathcal{U}_N^{\mathrm{reg}}(e)$ for $\varepsilon \sim p_0$, and let $L(x, c)$ be a differentiable design loss. We optimize

$$\min_{\varepsilon} \; L(D_e(T_\psi(\varepsilon; c, e)), c). \tag{21}$$

Because D_e maps every regular coordinate to $\mathcal{B}_N(e)$, each candidate scored during optimization satisfies the endpoint and link-length constraints. Cartesian optimization can instead lower the loss by leaving the feasible space, and that improvement may disappear when the chain is closed afterward. Section 3 tests this distinction directly.

2.5 Recursive compilation beyond chain geometry

To show how RCC applies outside chain geometry, we consider bounded fixed-total allocations. Consider distributing a total B among n variables with lower and upper bounds. For $n \geq 2$, finite bounds $\ell_i < u_i$, and $\sum_i \ell_i < B < \sum_i u_i$, the valid allocations are

$$\mathcal{A}_B = \{x \in \mathbb{R}^n : \ell_i < x_i < u_i, \; \sum_i x_i = B\}. \tag{22}$$

For a group of variables I, the rest of the tree only needs to know its subtotal. Let

$$L_I = \sum_{i \in I} \ell_i, \qquad U_I = \sum_{i \in I} u_i$$

be the smallest and largest subtotal that group can receive. Suppose a node with subtotal D is split into groups I_ℓ and I_r. The left subtotal must satisfy its own bounds while leaving a feasible subtotal for the right group, so

$$D_\ell \in (\alpha, \beta), \qquad \alpha = \max\{L_{I_\ell}, D - U_{I_r}\}, \qquad \beta = \min\{U_{I_\ell}, D - L_{I_r}\}. \tag{23}$$

A coordinate $t \in (0, 1)$ parameterizes this interval:

$$D_\ell = \alpha + t(\beta - \alpha), \qquad D_r = D - D_\ell.$$

Applying the same split to both children determines all x_i while preserving the total and every component bound.

Each internal split contributes one coordinate, giving $n - 1$ coordinates for $\mathcal{A}_B$. On a balanced tree, encoding and decoding require $O(n)$ work and $O(\log n)$ parallel depth. The same recursion also gives the volume factor. If $w_v = \beta_v - \alpha_v$ is the width of the feasible interval at node v, then

$$\mathcal{J}_B(t) = \sqrt{n} \prod_v w_v. \tag{24}$$

Appendix A.4 gives the derivation including the points where the active bounds change.

3 Experiments

We validate RCC coordinates and volume correction on protein backbones and bounded fixed-total allocations. We then use RCC to close samples from pretrained protein generators and measure the structural change introduced by enforcing the endpoint constraint. Finally, in CDR-H3 design, we test whether the same coordinates support conditional density modeling and feasible guidance and whether similar gains can be recovered by unconstrained optimization followed by repair.

3.1 RCC yields exact probabilistic coordinates across constraint families

We first verify that RCC provides exact coordinates for the constrained space and the correct volume correction for probability densities. Across protein bridges with $N = 2, \ldots, 31$, the RCC decoder satisfies the link-length and endpoint constraints to floating-point precision, the encoder recovers the original coordinates, and the RCC retraction fixes inputs that already satisfy the target constraints. On 240 prespecified configurations, the recursively computed log-volume factor agrees with the full decoder Jacobian and an independent action-angle chart. Its coordinate gradients also agree with autodifferentiated references with a maximum absolute discrepancy of 1.91×10^{-14}. These checks verify the coordinate identities and the volume correction used for density evaluation. Appendix C reports the full set of numerical checks.

To establish whether the construction extends beyond fixed-length chains, we repeat the coordinate and volume checks on 312 bounded fixed-total allocation instances with $n = 2, \ldots, 40$. Every decoded allocation satisfies its bounds and prescribed total, and the recursive volume and gradient calculations agree with full-Jacobian references to numerical precision. On 8,192 unseen predictions from a frozen model, RCC reduces the maximum total-budget error from 0.1915 to 7.11×10^{-15} without violating a component bound. Euclidean projection achieves slightly lower RMSE on the same predictions.

3.2 RCC enforces exact geometry with small changes to pretrained-generator outputs

We next test RCC on existing protein generators without retraining them or significantly altering their outputs. We evaluate identical generated outputs from LoopGen CDR generation, CATH fragment inpainting (Sillitoe et al., 2021), and FrameFlow motif-linker generation, holding the checkpoint,

Table 1: Structural preservation after closure of pretrained-generator outputs. For each setting, RCC and fixed-512 projection are applied to identical generated outputs. We report paired differences with 95% confidence intervals for the primary structural measure and structural diversity. RMSD, displacement, and diversity are reported in Å.

Setting	Parent units	Conditions / samples	Quality measure	Δ quality (95% CI)	Δ diversity (95% CI)	Max residual
LoopGen CDRs	191 PDBs	2,061 / 20,610	Best-of-10 Cα RMSD	$+0.00096$ ($+0.00010$, $+0.00180$)	-0.00023 (-0.00080, $+0.00030$)	5.96×10^{-8}
CATH fragments	256 domains	21,294 / 212,940	Mean Cα RMSD	$+0.00795$ ($+0.00748$, $+0.00842$)	$+0.01411$ ($+0.01337$, $+0.01484$)	5.96×10^{-8}
FrameFlow linkers	4 targets	73 / 292	Displacement from raw	$+0.00172$ (-0.00644, $+0.00790$)	$+0.00238$ (-0.00336, $+0.01087$)	6.27×10^{-7}

Table 2: Comparison of closure methods on 212,940 CATH generated outputs. All methods are evaluated on the same inputs and satisfy the common feasibility criterion. We report median/p99 pass counts and per-condition batch latency along with Cα RMSD, structural diversity, and clash rate.

Closure method	Passes p50/p99	Max residual ↓	Mean Cα RMSD ↓	Diversity ↑	Clash ↓	p99 ms ↓
Fixed-512 projection	512/512	1.94×10^{-6}	10.0237	9.4735	0.3282	38.485
Adaptive projection	16/512	5.01×10^{-5}	10.0237	9.4735	0.3282	2.113
FABRIK	1/2	2.53×10^{-5}	10.0256	9.4663	0.3442	**1.418**
FCCD-1	6/34	2.58×10^{-5}	**10.0118**	9.4579	0.3377	426.884
Riemannian Newton	2/3	2.55×10^{-5}	10.0242	9.4744	0.3283	1.741
Consensus ADMM	16/27	2.59×10^{-5}	10.0234	9.4730	0.3282	7.270
CoBarS	3/7	1.91×10^{-6}	10.0263	9.4804	0.3281	$19.444^{\dagger}$
Recursive compiler	**1/1 fixed**	1.94×10^{-6}	10.0317	**9.4877**	**0.3278**	2.674

[†]CoBarS timing includes external-process serialization and is not a kernel-level comparison.

conditioning, initial noise, and generated trajectory fixed within each comparison. This paired design isolates the effect of enforcing the terminal constraint.

Across 233,842 paired outputs, RCC returns a feasible backbone in every case. Relative to fixed-512 projection, the paired changes in the primary structural measures are all below 0.01 Å (Table 1). Each setting also satisfies its prespecified preservation criterion: the LoopGen difference remains below its preservation margin, every CATH length and endpoint-extension stratum passes, and the fixed FrameFlow motifs remain unchanged. Appendix D reports the full structural, diversity, clash, and sequence analyses.

We also compare RCC with seven alternative closure procedures on the same 212,940 CATH outputs. All eight methods satisfy the common feasibility criterion and produce similar structural summaries. RCC closes every chain in one fixed encode-decode pass, although FABRIK has the lowest p99 latency on this workload. The same RCC map also provides the probabilistic coordinates used below for density evaluation and guidance.

3.3 RCC ENABLES CONTEXT-CONDITIONED GENERATION AND GUIDANCE FOR CDR-H3 DESIGN

We assess whether RCC coordinates can support a learned conditional distribution over valid backbones and inference-time guidance. We evaluate this prospectively on a PDB-disjoint panel of antibody-antigen complexes containing 288 eligible design conditions from 255 PDB structures, with CDR-H3 lengths from 6 to 26 residues. For each condition, the model receives the fixed anchors, antibody framework, epitope geometry, and H3 length, but not the native H3 coordinates or sequence. Candidates are scored using epitope contacts, steric clashes, local backbone geometry, and exact closure; native RMSD is used only for retrospective evaluation.

We first test whether the learned coordinate density depends on the design context. The model assigns native H3 loops higher density than a uniform coordinate distribution in 92.0% of conditions. To test whether this preference reflects the surrounding antibody-antigen context rather than only loop length and endpoint separation, we permute contexts across PDBs. This lowers the mean native-loop log density by 1.152 nats, and the matched context is preferred in 61.3% of conditions. The decrease under context permutation indicates that the learned distribution uses the surrounding antibody-antigen context.

Table 3: CDR-H3 inference in RCC coordinates. All policies are evaluated on 288 conditions from 255 PDB entries with a budget of 64 feasible objective evaluations per condition. The first three rows are prospective and the remaining rows are post hoc controls. C_α clean reports the fraction without nonlocal C_α clashes. Diversity and native RMSD are reported in Å.

Evidence	Inference policy	Objective ↓	Success ↑	Cα clean ↑	Diversity ↑	Native Cα RMSD ↓
Prospective	Uniform IID	0.59156	2.78%	28.13%	11.102	9.063
Prospective	Learned IID	0.10703	42.01%	81.60%	5.343	5.564
Prospective	RCC-coordinate guidance (Adam)	**0.01653**	**70.14%**	**98.61%**	5.841	6.008
Post hoc	CEM	0.05460	42.71%	87.50%	3.656	5.797
Post hoc	CMA-ES	0.06027	44.10%	87.15%	4.186	5.806
Post hoc	UCT chart tree	0.23420	35.42%	72.57%	3.914	5.410
Post hoc	RCC-coordinate guidance (Langevin)	0.01849	72.22%	98.26%	3.143	5.537

We next compare sampling and guidance under a fixed inference budget. With 64 feasible objective evaluations per condition, uniform sampling from RCC coordinates succeeds on 2.78% of conditions. Sampling from the learned conditional distribution raises success to 42.01%, so the learned proposal places more probability on backbones that satisfy the reported structural criteria. Starting from the same learned distribution and optimizing in RCC coordinates raises success further to 70.14%. The guided samples also improve the design objective and clash rate (Table 3), while all 18,432 backbones evaluated during guidance satisfy the endpoint and link-length constraints.

To test whether the improvement is specific to Adam, we compare it with CEM, CMA-ES, UCT, and score-guided Langevin under the same evaluation budget. CEM, CMA-ES, and UCT have higher objective values and lower structural success than Adam, while Langevin performs comparably to Adam. The advantage over the tested gradient-free methods persists across budgets from 8 to 128 evaluations (Appendix E.3). We next test whether unconstrained optimization followed by repair can recover the same gains.

3.4 Feasible guidance outperforms unconstrained guidance followed by repair

We evaluate whether feasibility must be maintained during search or can instead be imposed only on the final candidates. We hold the starting samples and evaluation budget fixed and change only the space in which optimization is performed.

In this post-hoc comparison, we repeat the 15-step search from the same four initial samples and use the same 64 scored-objective evaluations per condition as RCC guidance. Instead of optimizing RCC coordinates, we allow each link vector to move independently in Cartesian space. The unconstrained search reaches a lower mean objective of 0.00755, compared with 0.01653 for RCC guidance, but leaves the feasible backbone space, with a maximum normalized link error of 3.41. We then close each final Cartesian candidate using adaptive projection, FABRIK, CoBarS, or the RCC retraction.

All four closure methods restore geometric validity, but the objective gains from unconstrained optimization do not survive repair. After closure, the mean objective rises to 0.547-0.690, and structural success is 6.6%-11.1%, compared with 70.1% when guidance is performed directly in RCC coordinates (Table 4). The same degradation occurs with all four terminal maps, including the RCC retraction itself, showing that the effect is not specific to a particular repair algorithm. Instead, the Cartesian optimizer is able to exploit geometries that are unavailable once the required backbone constraints are restored.

Table 4: Guidance in RCC coordinates compared with Cartesian guidance followed by terminal closure. All methods use the same initial samples and 64 objective evaluations per condition. Every closure method returns a valid backbone, but only RCC guidance maintains feasibility throughout optimization.

Search and terminal map	Exact during search	Exact output	Objective ↓	Success ↑	Cα clean ↑
RCC-coordinate guidance (Adam)	Yes	Yes	**0.0165**	**70.1%**	**98.6%**
Cartesian, before repair	No	No	0.0075	0.0%	100.0%
Cartesian + adaptive projection	No	Yes	0.6497	7.6%	48.6%
Cartesian + FABRIK	No	Yes	0.6701	6.6%	45.5%
Cartesian + CoBarS	No	Yes	0.6901	6.9%	44.8%
Cartesian + recursive retraction	No	Yes	0.5472	11.1%	53.5%

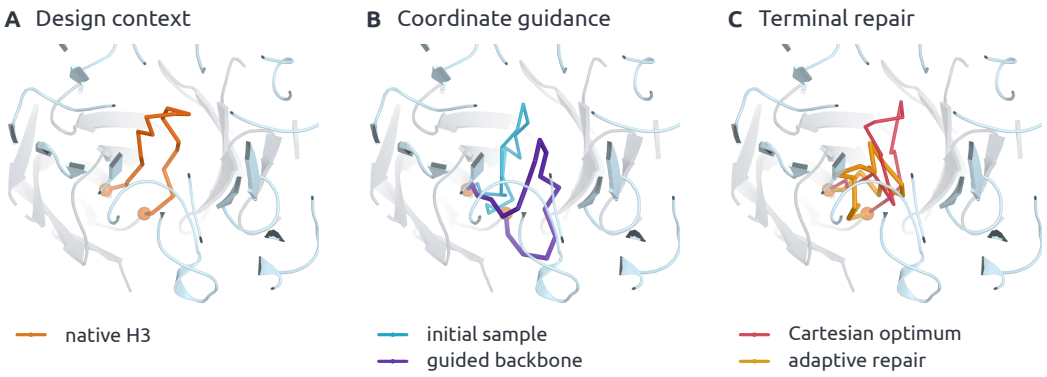


Figure 3: CDR-H3 guidance and terminal repair. (A) Design context and native H3, which is hidden during inference. (B) Initial sample and backbone after RCC-coordinate guidance. (C) Cartesian-guided backbone before and after adaptive repair, illustrating how terminal closure can alter a candidate found outside the feasible backbone space.

Cartesian optimization can lower the objective by moving through invalid geometries, but those gains disappear when the final backbone must satisfy the endpoint and link-length constraints. RCC instead evaluates every candidate under those constraints during search. Figure 3 shows a Cartesian candidate whose geometry changes substantially during repair. Appendix E.5 compares RCC-coordinate guidance with alternative methods that also remain feasible throughout optimization.

4 Discussion

RCC shows that hard geometric constraints can define the space in which protein backbones are generated and optimized rather than serving only as a final repair step. RCC can be applied to pretrained generators without retraining and supports probability modeling and guidance over feasible backbones. In the CDR-H3 experiment, every terminal repair method restores feasibility after unconstrained optimization, but none preserves the objective gains achieved during search. RCC instead searches over feasible backbones from the outset, ensuring that the optimizer evaluates every candidate under the constraints that the final design must satisfy.

RCC is not intended to replace specialized closure or feasibility-maintaining optimization methods. Instead, it provides a recursive coordinate system that enforces the hard constraint, defines normalized densities, and supports guidance without retraining the underlying generator. In post-hoc development comparisons, RCC-coordinate guidance and the strongest feasibility-maintaining controls, including Riemannian guidance, perform comparably. These results therefore support the benefit of maintaining feasibility during search rather than a general optimization advantage of RCC coordinates. The bounded-allocation experiment shows that RCC also applies when recursive subproblems can be summarized by compact boundary data and assembled through invertible local compatibility maps. For proteins, the current framework makes only the fixed endpoints and C_α link lengths exact. Extending RCC to broader protein-design applications will require incorporating richer structural constraints and their corresponding local parameterizations. Establishing the biological relevance and utility of these designs will require experimental validation.

AI USE STATEMENT

Generative AI tools assisted with conceptual and mathematical development, software implementation and debugging, experimental design and diagnostic analysis, interpretation of computational results, literature search and summarization, and drafting and editing the manuscript. The authors are responsible for the final code, reported results, proofs, citations and conclusions, including material produced with AI assistance.

ETHICS STATEMENT

This work studies constrained protein backbone generation and optimization computationally and does not include human or animal experiments. Protein design methods can be misused or produce structures that appear plausible without being biologically functional. Exact geometric constraints, model scores, sequence compatibility and structure prediction do not establish folding, binding, safety or biological activity. Generated structures therefore remain computational designs that require experimental validation before biological use.

REPRODUCIBILITY STATEMENT

The Methods and Appendix specify the recursive constraint construction, fixed-endpoint backbone coordinates, intrinsic volume calculation, pretrained generator interfaces, guidance procedures, benchmark protocols and evaluation metrics. The Appendix provides the mathematical proofs, numerical validation, experimental protocols and complete supporting results. Randomization, evaluation budgets, dataset splits and prospective and post hoc analyses are reported with the corresponding experiments.

REFERENCES

Brennan Abanades, Wing Ki Wong, Fergus Boyles, Guy Georges, Alexander Bujotzek, and Charlotte M. Deane. ImmuneBuilder: Deep-learning models for predicting the structures of immune proteins. *Communications Biology*, 6(1):575, 2023. doi: 10.1038/s42003-023-04927-7.

Mansoor Ahmed, Nadeem Taj, Imdad Ullah Khan, Hemanth Venkateswara, and Murray Patterson. CHIMERA-bench: A benchmark dataset for epitope-specific antibody design. In *ICLR Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2026. URL https://openreview.net/forum?id=PyZvVIJbSy.

John D. Boom, Matthew Greenig, Pietro Sormanni, and Pietro Liò. Score-based generative models for designing binding peptide backbones. *arXiv preprint arXiv:2310.07051*, 2023. URL https://arxiv.org/abs/2310.07051.

Ricky T. Q. Chen and Yaron Lipman. Flow matching on general geometries. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=g7ohDlTITL.

Jacob K. Christopher, Austin Seamann, Jingyi Cui, Sagar D. Khare, and Ferdinando Fioretto. Constrained diffusion for protein design with hard structural constraints. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=kkvqVRu2Zy.

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Alexis Courbet, Rob J. de Haas, Neville Bethel, Philip J. Y. Leung, Timothy F. Huddy, Sam Pellock, Doug Tischer, Frederick Chan, Brian Koepnick, Hanneke 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.

Valentin De Bortoli, Emile Mathieu, Michael Hutchinson, James Thornton, Yee Whye Teh, and Arnaud Doucet. Riemannian score-based generative modelling. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2202.02763.

Berthy T. Feng, Ricardo Baptista, and Katherine L. Bouman. Neural approximate mirror maps for constrained diffusion models. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=vgZDcUetWS.

Nikolaus Hansen. The CMA evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016. doi: 10.48550/arXiv.1604.00772.

Chin-Wei Huang, Milad Aghajohari, Avishek Joey Bose, Prakash Panangaden, and Aaron Courville. Riemannian diffusion models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2208.07949.

Henry Kenlay, Frédéric A. Dreyer, Daniel Cutting, Daniel Nissley, and Charlotte M. Deane. ABodyBuilder3: Improved and scalable antibody structure predictions. *Bioinformatics*, 40(10):btae576, 2024. doi: 10.1093/bioinformatics/btae576.

Xinpeng Li, Enming Liang, and Minghua Chen. Gauge flow matching: Efficient constrained generative modeling over general convex set and beyond. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/9304fe9848ca4db73c79ac0bac414aef-Abstract-Conference.html.

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

Piotr Rotkiewicz and Jeffrey Skolnick. Fast procedure for reconstruction of full-atom protein models from reduced representations. *Journal of Computational Chemistry*, 29(9):1460–1465, 2008. doi: 10.1002/jcc.20906.

Ian Sillitoe, Nicola Bordin, Natalie Dawson, Vaishali P. Waman, Paul Ashford, Harry M. Scholes, Camilla S. M. Pang, Laurel Woodridge, Clemens Rauer, Neeladri Sen, Mahnaz Abbasian, Sean Le Cornu, Su Datt Lam, Karel Berka, Ivana Hutařová Varekova, Radka Svobodová, Jon Lees, and Christine A. Orengo. CATH: Increased structural coverage of functional space. *Nucleic Acids Research*, 49(D1):D266–D273, 2021. doi: 10.1093/nar/gkaa1079.

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

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

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, 2023b. URL https://proceedings.mlr.press/v202/yim23a.html.

APPENDIX

A THEORY

The appendix first proves the recursive encoder-decoder construction, then derives the fixed-endpoint backbone coordinates and their volume factor followed by the bounded fixed-total allocation construction.

Metric and chart conventions. For protein backbones, we measure geometry directly in link-vector coordinates. If $v, w \in (\mathbb{R}^d)^N$ are tangent vectors, we use

$$\langle v, w \rangle = \sum_i v_i^\top w_i. \tag{25}$$

This choice fixes the intrinsic volume measure used in all of the bridge density calculations below.

We represent each directional degree of freedom with a local one-to-one chart on the sphere. When the direction space is $\mathbb{S}^0 = \{-1, +1\}$, the direction is simply a discrete branch choice and contributes no continuous coordinates. We adopt the convention that the determinant of a 0×0 matrix is 1, which lets the same volume formulas cover this case. All derivatives below are taken on chart pieces where the corresponding map is well defined and full rank.

A.1 RECURSIVE CHART ASSEMBLY

To prove Theorem 2.2, fix a rooted binary tree and an ordered decomposition $x_v = x_\ell \star x_r$ at every internal node, so that the two children can be recovered from x_v. We use the same local chart and branch choices throughout the tree.

For a node v and summary s, let

$$\mathcal{X}_v(s) = \{x_v : H_v(x_v) = s\}.$$

At a leaf, choose a regular coordinate domain $U_v^{\mathrm{reg}}(s)$ and inverse maps

$$D_{v,s} : U_v^{\mathrm{reg}}(s) \to \mathcal{X}_v(s), \qquad E_{v,s} : D_{v,s}(U_v^{\mathrm{reg}}(s)) \to U_v^{\mathrm{reg}}(s).$$

At an internal node, a local coordinate z_v determines the child summaries through

$$(s_\ell, s_r) = g_{v,s}(z_v).$$

The remaining coordinates belong to the corresponding child domains, so $U_v^{\mathrm{reg}}(s)$ consists of tuples (z_v, u_ℓ, u_r) with $u_\ell \in U_\ell^{\mathrm{reg}}(s_\ell)$ and $u_r \in U_r^{\mathrm{reg}}(s_r)$. The decoder applies the two child decoders and joins their outputs:

$$D_{v,s}(z_v, u_\ell, u_r) = D_{\ell,s_\ell}(u_\ell) \star D_{r,s_r}(u_r). \tag{26}$$

We define $\mathcal{X}_v^{\mathrm{reg}}(s)$ as the image of this map. For $x_v = x_\ell \star x_r$ in that image, the encoder first computes $s_\ell = H_\ell(x_\ell)$ and $s_r = H_r(x_r)$, then reverses the same split:

$$E_{v,s}(x_v) = \big(a_{v,s}(s_\ell, s_r), E_{\ell,s_\ell}(x_\ell), E_{r,s_r}(x_r)\big). \tag{27}$$

Proof of Theorem 2.2. We prove by induction that decoding realizes the summary assigned to every node and that the encoder and decoder are mutually inverse on the regular domains above. Both statements hold at a leaf by assumption.

Consider an internal node v with assigned summary s, and suppose the claims hold for its two children. For coordinates $u = (z_v, u_\ell, u_r)$, let $(s_\ell, s_r) = g_{v,s}(z_v)$. Since these child summaries are compatible, $m_v(s_\ell, s_r) = s$. By the induction hypothesis, the child decoders realize s_ℓ and s_r, so

$$H_v(D_{v,s}(u)) = m_v(H_\ell(D_{\ell,s_\ell}(u_\ell)), H_r(D_{r,s_r}(u_r))) \tag{28}$$

$$= m_v(s_\ell, s_r) = s. \tag{29}$$

Thus the parent decoder realizes its assigned summary.

Encoding this decoded configuration recovers the same child summaries. Because $a_{v,s}$ is the inverse of $g_{v,s}$, it recovers z_v, and the induction hypothesis recovers u_ℓ and u_r. Hence

$$E_{v,s}(D_{v,s}(u)) = u.$$

For the reverse identity, take $x_v = x_\ell \star x_r \in \mathcal{X}_v^{\mathrm{reg}}(s)$ and compute its child summaries s_ℓ and s_r. The encoder records $z_v = a_{v,s}(s_\ell, s_r)$, so decoding this coordinate returns the same pair (s_ℓ, s_r). The induction hypothesis then reconstructs both child configurations, giving

$$D_{v,s}(E_{v,s}(x_v)) = x_v.$$

Applying the induction at the root gives $H_{\mathrm{root}} = F = s$ and proves both inverse identities for the assembled maps.

A full binary tree with n leaves has $2n-1$ nodes. Encoding and decoding visit each node once, so $O(k)$ work per node gives $O(nk)$ total work. Nodes at the same level can be processed in parallel, and a balanced tree has $O(\log n)$ levels, giving $O(k \log n)$ parallel depth. □

The volume calculation below differentiates these assembled maps. If the local maps are C^1 on the chosen chart pieces, then the assembled encoder and decoder are also C^1 there by composition.

A.2 Fixed-endpoint bridge coordinates

The recursive construction ends at chains of two or three links. We first characterize their reachable endpoints and the number of degrees of freedom remaining after the endpoint is fixed.

For n links of length b, define the endpoint map

$$\bar{F}_n(x_1, \dots, x_n) = \sum_{i=1}^{n} x_i, \qquad (x_1, \dots, x_n) \in (\mathbb{S}_b^{d-1})^n,$$

and let

$$\mathcal{B}_n(D) = \bar{F}_n^{-1}(D)$$

denote the set of chains with endpoint displacement D.

Lemma A.1 (Reachability and fixed-endpoint dimension)**.** *For $b > 0$, $d \geq 2$, and $n \geq 2$, the reachable endpoint displacements are exactly $\overline{B}(0, nb)$. At any configuration whose links are not all collinear, $d\bar{F}_n$ has rank d, and $\mathcal{B}_n(D)$ is locally a smooth manifold of dimension*

$$n(d-1) - d.$$

Proof. The triangle inequality gives $\|\sum_i x_i\| \leq nb$, so no chain can reach outside $\overline{B}(0, nb)$. We show by induction that every displacement inside this ball can be reached.

For two links, let $0 < \|D\| < 2b$. Choose $z \in D^\perp$ with $\|z\|^2 = b^2 - \|D\|^2/4$ and set

$$x_1 = \frac{D}{2} + z, \qquad x_2 = \frac{D}{2} - z. \tag{30}$$

Both links have length b and sum to D. The case $D = 0$ is obtained with antiparallel links, and $\|D\| = 2b$ with parallel links. Hence every point in $\overline{B}(0, 2b)$ is reachable.

Now suppose the claim holds for $n-1$ links and take $\|D\| \leq nb$. If $D \neq 0$, choose the first link in the direction of D,

$$x_1 = b\frac{D}{\|D\|}.$$

The remaining displacement has length $\big|\|D\| - b\big| \leq (n-1)b$, so the remaining links exist by the induction hypothesis. If $D = 0$, choose any first link. The remaining links must span $-x_1$, whose length is b, and again exist by induction. This proves the reachability claim.

We next compute the dimension of a regular fiber. At $x = (x_1, \dots, x_n)$,

$$T_x(\mathbb{S}_b^{d-1})^n = \{(v_1, \dots, v_n) : v_i \perp x_i\}, \qquad d\bar{F}_n(v) = \sum_{i=1}^{n} v_i.$$

Suppose $a \in \mathbb{R}^d$ is orthogonal to the image of $d\bar{F}_n$. Varying one v_i at a time shows that a is orthogonal to $x_i^\perp$, so a must be parallel to every x_i. If the links are not all collinear, the only such vector is $a = 0$. Hence $d\bar{F}_n$ is surjective and has rank d.

Since $(\mathbb{S}_b^{d-1})^n$ has dimension $n(d-1)$, the implicit function theorem gives

$$\dim \mathcal{B}_n(D) = n(d-1) - d.$$

□

We now construct the two- and three-link charts with exactly these dimensions.

A.2.1 TWO-LINK COORDINATES

For a two-link leaf, fixing the endpoint displacement D fixes the midpoint of the two links. The only remaining freedom is the direction in which the links bend away from D. Assume $0 < \|D\| < 2b$ and write $x_1 = D/2 + z$ and $x_2 = D/2 - z$. Requiring both links to have length b gives $D^\top z = 0$ and $\|z\|^2 = b^2 - \|D\|^2/4$. Thus z lies perpendicular to D and has fixed length.

Let

$$h = \sqrt{b^2 - \frac{\|D\|^2}{4}}. \tag{31}$$

Every two-link bridge therefore has the form

$$x_1 = \frac{D}{2} + hv, \qquad x_2 = \frac{D}{2} - hv, \qquad v \in D^\perp, \quad \|v\| = 1. \tag{32}$$

Since $0 < \|D\| < 2b$, we have $h > 0$, and the inverse recovers $v = (x_1 - x_2)/(2h)$.

The unit vectors in $D^\perp$ form $\mathbb{S}^{d-2}$, giving the $d-2$ continuous coordinates predicted by Lemma A.1. An orthonormal frame for $D^\perp$ provides local coordinates on this sphere. When $d = 2$, $\mathbb{S}^0 = \{-1, +1\}$ contains only two discrete choices, so the chart has no continuous coordinate.

A.2.2 THREE-LINK COORDINATES

For a three-link leaf, we choose the first link and then use the two-link construction for the remaining pair. Write $x_1 = bu$ with $u \in \mathbb{S}^{d-1}$. The last two links must span $R = D - bu$, so their regular chart requires $0 < \|R\| < 2b$.

We first determine which directions u satisfy this condition. Let $L = \|D\|$. When $L > 0$, write $a = D/L$. The upper bound $\|D - bu\| < 2b$ is equivalent to

$$u^\top a > \tau(L), \qquad \tau(L) = \frac{L^2 - 3b^2}{2bL}. \tag{33}$$

If $L < b$, every direction is allowed. When $L = b$, the direction $u = a$ gives $R = 0$, and $u = -a$ gives the fully extended case $\|R\| = 2b$. For $b < L < 3b$, the valid directions form an open spherical cap centered at a.

To keep the coordinate domain fixed as D changes, we map a reference direction $w \in \mathbb{S}^{d-1}$ into this cap. When $L \le b$, no remapping is needed on the regular part of the sphere, so we take $u = w$. For $b < L < 3b$, work in a local sphere chart away from the axial poles and write

$$q_0 = w^\top a, \qquad v = \frac{w - q_0 a}{\sqrt{1 - q_0^2}}, \tag{34}$$

$$q = \tau(L) + \frac{q_0 + 1}{2}\big(1 - \tau(L)\big), \qquad u = qa + \sqrt{1 - q^2}\, v. \tag{35}$$

The scalar map sends $q_0 \in (-1, 1)$ to $q \in (\tau(L), 1)$ and leaves the transverse direction v unchanged. Hence u lies in the admissible cap.

The inverse simply reverses this rescaling. Given u in the cap,

$$q = u^\top a, \qquad v = \frac{u - qa}{\sqrt{1 - q^2}}, \tag{36}$$

$$q_0 = \frac{2(q - \tau(L))}{1 - \tau(L)} - 1, \qquad w = q_0 a + \sqrt{1 - q_0^2}\, v. \tag{37}$$

The forward and inverse maps are smooth on the chosen sphere chart. Neighboring charts cover the axial directions.

Once the first link is fixed, the remaining pair is exactly the two-link problem above. For $0 < \|R\| < 2b$, let

$$m = \frac{R}{2}, \qquad h = \sqrt{b^2 - \frac{\|R\|^2}{4}},$$

and choose a unit vector $\omega \in R^\perp$. Then

$$x_2 = m + h\omega, \qquad x_3 = m - h\omega. \tag{38}$$

Both links have length b, and together with x_1 they sum to D. The first-link direction contributes $d-1$ coordinates and ω contributes $d-2$, giving the required total of $2d-3$. The inverse recovers $u = x_1/b$, then w from the cap map, and finally $\omega = (x_2 - x_3)/(2h)$.

The regular chart excludes the residual cases $R = 0$ and $\|R\| = 2b$. When $R = 0$, the last two links form an antiparallel pair parameterized by $v \in \mathbb{S}^{d-1}$. When $\|R\| = 2b$, the pair is fully extended and $h = 0$. We treat these boundary cases with separate terminal charts.

A.2.3 Internal lens coordinates

Consider an internal node whose left and right children contain n_ℓ and n_r links. Their reach radii are $R_\ell = n_\ell b$ and $R_r = n_r b$. We choose the point

$$c = \frac{R_\ell}{R_\ell + R_r} D$$

inside the lens. Since $\|D\| < R_\ell + R_r$,

$$\frac{\|c\|}{R_\ell} = \frac{\|c - D\|}{R_r} = \frac{\|D\|}{R_\ell + R_r} < 1,$$

so c lies strictly inside both child reachability balls.

Starting from c, each direction $u \in \mathbb{S}^{d-1}$ remains inside the lens until it reaches the boundary of one of these balls. Solving for the two ray intersections gives

$$\rho_\ell(u) = -c^\top u + \sqrt{(c^\top u)^2 + R_\ell^2 - \|c\|^2}, \tag{39}$$

$$\rho_r(u) = -(c - D)^\top u + \sqrt{((c - D)^\top u)^2 + R_r^2 - \|c - D\|^2}. \tag{40}$$

The first boundary encountered along the ray is therefore $\rho_D(u) = \min\{\rho_\ell(u), \rho_r(u)\}$.

A direction u and radial fraction $r \in (0, 1)$ now determine the meeting point

$$X = c + r\rho_D(u)u. \tag{41}$$

Because $r < 1$, X lies strictly inside both reachability balls. By Lemma A.1, both child chains can therefore realize the displacements X and $D - X$.

The map is inverted directly from the meeting point. For $X \neq c$,

$$u = \frac{X - c}{\|X - c\|}, \qquad r = \frac{\|X - c\|}{\rho_D(u)}. \tag{42}$$

An interior point satisfies $0 < r < 1$, so these formulas recover exactly the direction and radial fraction used to construct X.

While the same ball determines the first exit, ρ_D and the lens map are smooth. At a tie between the two exit distances, the map remains invertible but is only piecewise smooth. The derivative calculations below stay within a single smooth branch.

A.2.4 Proof of Proposition 2.3

Proof. Fix the balanced recursion tree and the local charts used by the encoder and decoder, and work on $\mathcal{U}_N^{\mathrm{reg}}(e)$, where each lens map and terminal chart is well defined and invertible.

We first prove feasibility by induction over the tree. The two- and three-link constructions above satisfy their link-length and endpoint constraints directly. At an internal node with displacement D, the lens coordinates choose a meeting point X with

$$\|X\| < n_\ell b, \qquad \|D - X\| < n_r b.$$

Lemma A.1 therefore guarantees that both child displacements are reachable. The child decoders produce chains spanning X and $D - X$, and concatenating them gives a chain spanning D. Applying the same argument to the root shows that every decoded link has length b and that the complete chain sums to e. Hence $D_e(u) \in \mathcal{B}_N(e)$.

The inverse follows from the same recursion. For a decoded bridge, summing the links in the left child recovers the meeting point X. The inverse lens map recovers its direction and radial coordinate, and the terminal inverses recover the leaf coordinates. Repeating this at every node gives

$$E_e(D_e(u)) = u.$$

Conversely, a bridge in the chosen chart determines the same meeting point at each node from the sum of its left-child links. Encoding and decoding that point returns the same split, and the terminal encoder-decoder pairs recover the original links. Induction over the tree gives

$$D_e(E_e(x)) = x.$$

□

The coordinate count also follows directly from the tree. Suppose there are p two-link leaves and t three-link leaves. Then $2p + 3t = N$, and the tree has $p + t - 1$ internal nodes. Each internal lens contributes d coordinates, each two-link leaf contributes $d - 2$, and each three-link leaf contributes $2d - 3$. The total is therefore

$$d(p + t - 1) + (d - 2)p + (2d - 3)t = N(d - 1) - d, \tag{43}$$

which matches the dimension in Lemma A.1. Discrete branch choices in $d = 2$ add no continuous dimensions.

A noncollinear bridge can still pass through a singular intermediate split. For example, take $d = 2$, $N = 4$, $b = 1$, and $e = 0$, and let

$$x(\theta, \phi) = \big(v(\theta), -v(\theta), w(\phi), -w(\phi)\big),$$

where $v(\theta) = (\cos\theta, \sin\theta)$ and $w(\phi) = (\cos\phi, \sin\phi)$ are not parallel. As θ and ϕ vary, these form a two-dimensional family of regular bridges. Under the balanced split after the first two links, however, the meeting point is $X = 0 = c$, where the lens direction is undefined. This is why Proposition 2.3 is stated on the chosen regular chart domain rather than on all noncollinear bridges at once.

Corollary A.2 (Bridge retraction)**.** *Let Ω_e be the set of ambient inputs for which the normalization and the source and target chart maps in R_e are well defined. Then $R_e(y) \in \mathcal{B}_N(e)$ for every $y \in \Omega_e$, and*

$$R_e(x) = x \qquad \textit{for every } x \in \Omega_e \cap \mathcal{B}_N(e).$$

Proof. The normalization gives a chain $q = (q_1, \ldots, q_N)$ with $\|q_i\| = b$ and endpoint displacement $e_y = \sum_i q_i$. Hence $q \in \mathcal{B}_N(e_y)$, and Proposition 2.3 gives

$$R_e(y) = D_e(E_{e_y}(q)) \in \mathcal{B}_N(e).$$

If $x \in \mathcal{B}_N(e)$, then $\sum_i x_i = e$, so $\delta = 0$. Since each link already has length b, the normalization leaves the chain unchanged: $q = x$ and $e_y = e$. Encoder-decoder inversion then gives

$$R_e(x) = D_e(E_e(x)) = x.$$

□

A.2.5 RECURSIVE BRIDGE COMPILER

Algorithm 1 gives the bridge compiler used in the experiments. It first converts an ambient chain into an equal-link source bridge, encodes that bridge in recursive coordinates, and then decodes the same coordinates under the required endpoint displacement. The recursion stops at the two- and three-link charts derived above.

Algorithm 1 Recursive constraint compilation for a fixed-endpoint bridge

Require: links $y_{1:N} \in (\mathbb{R}^d)^N$, target displacement e, link length b, and $\|e\| < Nb$
Ensure: a bridge in $\mathcal{B}_N(e)$, or a regularity exception

$\delta \leftarrow (e - \sum_j y_j)/N$
$z_i \leftarrow y_i + \delta$ for $i = 1, \dots, N$
if $\min_i \|z_i\| = 0$ **then**
    return REGULARITYEXCEPTION
$q_i \leftarrow b z_i / \|z_i\|$ for $i = 1, \dots, N$
$P_0 \leftarrow 0$ and $P_k \leftarrow \sum_{i=1}^{k} q_i$ for $k = 1, \dots, N$
$u \leftarrow$ ENCODEBRIDGE$(q, P, 1, N)$
return DECODEBRIDGE(e, N, u)
procedure ENCODEBRIDGE(x, P, a, n)
    $D \leftarrow P_{a+n-1} - P_{a-1}$
    if $n \le 3$ **then**
        return TERMINALENC$(x_{a:a+n-1}, D, b)$
    $n_\ell \leftarrow \lfloor n/2 \rfloor$ and $n_r \leftarrow n - n_\ell$
    $X \leftarrow P_{a+n_\ell-1} - P_{a-1}$
    $c \leftarrow (n_\ell/n)D$
    if $X = c$ **then**
        return REGULARITYEXCEPTION
    $v \leftarrow (X - c)/\|X - c\|$
    $r \leftarrow \|X - c\|/\rho_D(v; n_\ell, n_r)$
    $u_\ell \leftarrow$ ENCODEBRIDGE(x, P, a, n_ℓ)
    $u_r \leftarrow$ ENCODEBRIDGE$(x, P, a + n_\ell, n_r)$
    return (v, r, u_ℓ, u_r)
procedure DECODEBRIDGE(D, n, u)
    if $n \le 3$ **then**
        return TERMINALDEC(D, b, u)
    parse $u = (v, r, u_\ell, u_r)$
    $n_\ell \leftarrow \lfloor n/2 \rfloor$ and $n_r \leftarrow n - n_\ell$
    $c \leftarrow (n_\ell/n)D$
    $X \leftarrow c + r\rho_D(v; n_\ell, n_r)v$
    $x_\ell \leftarrow$ DECODEBRIDGE(X, n_ℓ, u_ℓ)
    $x_r \leftarrow$ DECODEBRIDGE$(D - X, n_r, u_r)$
    return $x_\ell \,\|\, x_r$

The prefix sums let the encoder obtain each subchain displacement without summing the same links again. After that, encoding and decoding each visit every node once and use $O(d)$ work per node, giving $O(Nd)$ total work. The two children of a split are independent once their endpoint displacements are known, so the recursion has $O(\log N)$ depth on a balanced tree for fixed d.

The implementation checks the same chart conditions used above and stops if one of them fails. Our experiments evaluate the recursion serially, so the $O(\log N)$ depth describes available parallelism rather than measured runtime.

A.3 RECURSIVE BRIDGE VOLUME

At an internal split, moving the meeting point changes the endpoint of each child chain. To compute how this affects volume, we first determine the smallest tangent motion of a child chain that produces a given endpoint displacement.

A.3.1 ENDPOINT MOTION

For a subtree I, let

$$\mathcal{E}_I = (\mathbb{S}_b^{d-1})^{|I|}, \qquad \bar{F}_I(x) = \sum_{i \in I} x_i \tag{44}$$

denote its link space and endpoint map. Write $\widehat{x}_i = x_i/b$ and $P_i = I_d - \widehat{x}_i \widehat{x}_i^\top$, the projection onto the tangent space of the sphere at x_i.

If $v = (v_i)_{i \in I}$ is a tangent perturbation, then

$$D\bar{F}_I(x)v = \sum_{i \in I} v_i, \qquad D\bar{F}_I(x)^* a = (P_i a)_{i \in I}. \tag{45}$$

It follows that

$$M_I = D\bar{F}_I(x) D\bar{F}_I(x)^* = \sum_{i \in I} P_i. \tag{46}$$

For any $a \in \mathbb{R}^d$,

$$a^\top M_I a = \sum_{i \in I} \|P_i a\|^2. \tag{47}$$

This is zero only when a is parallel to every link. Hence M_I is positive definite at every noncollinear configuration.

Lemma A.3 (Minimum-norm endpoint motion)**.** *At a noncollinear configuration, the minimum-norm tangent perturbation that changes the endpoint by δs is*

$$K_I \delta s, \qquad K_I = D\bar{F}_I(x)^* M_I^{-1}. \tag{48}$$

Its squared norm is $\delta s^\top M_I^{-1} \delta s$.

Proof. Let $L = D\bar{F}_I(x)$. Since $LL^* = M_I$,

$$LK_I = LL^* M_I^{-1} = I_d, \tag{49}$$

so $K_I \delta s$ produces the required endpoint displacement.

The vector $K_I \delta s$ lies in the image of L^* and is therefore orthogonal to $\ker L$. Any other tangent perturbation with the same endpoint change can be written as $K_I \delta s + w$ for some $w \in \ker L$. Therefore

$$\|K_I \delta s + w\|^2 = \|K_I \delta s\|^2 + \|w\|^2, \tag{50}$$

so $K_I \delta s$ is the unique minimum-norm choice. Finally, $K_I^* K_I = M_I^{-1}$, which gives the stated squared norm. □

A.3.2 MEETING-POINT JACOBIAN

We next compute the Jacobian of the lens coordinates. Let $\xi \in \mathbb{R}^{d-1}$ be local coordinates for the direction $u(\xi)$, with radial coordinate $r \in (0, 1)$. The meeting point is

$$X = c + r\rho_D(u)u. \tag{51}$$

With D fixed, its derivatives are

$$\partial_r X = \rho_D(u)u, \tag{52}$$

$$\partial_{\xi_j} X = r\, \partial_{\xi_j} \rho_D(u)\, u + r\rho_D(u)\, \partial_{\xi_j} u. \tag{53}$$

The first term in each angular derivative points in the same direction as the radial derivative. Subtracting the corresponding multiple of the radial column therefore leaves the determinant unchanged and reduces the angular columns to $r\rho_D(u)\partial_{\xi_j} u$. Since $u^\top \partial_{\xi_j} u = 0$, these columns are perpendicular to the radial column. If

$$G_u = (\partial_\xi u)^\top (\partial_\xi u)$$

is the metric of the local sphere chart, we obtain

$$|\det B_v| = r^{d-1} \rho_D(u)^d \sqrt{\det G_u}. \tag{54}$$

The factor $\sqrt{\det G_u}$ is the volume element of the local sphere chart. This calculation applies on each smooth branch of ρ_D; at an exit tie, the lens map is only piecewise smooth.

A.3.3 Proof of Theorem 2.4

Proof. Consider an internal node with left and right child decoders $D_\ell(X, z_\ell)$ and $D_r(D - X, z_r)$. Let

$$J_\ell = \partial_{z_\ell} D_\ell, \qquad J_r = \partial_{z_r} D_r$$

denote derivatives with respect to the child coordinates, and let $A_\ell = \partial_X D_\ell$ and $A_r = \partial_{D-X} D_r$ describe how the children move when their endpoints move.

Because each child decoder realizes its requested endpoint, $D\bar{F}_C A_C = I_d$ for $C \in \{\ell, r\}$. The columns of J_C span motions that leave that endpoint fixed. If

$$\Pi_C = J_C (J_C^\top J_C)^{-1} J_C^\top$$

is the orthogonal projector onto these motions, then $A_C - K_C$ is endpoint preserving, where K_C is the minimum-norm endpoint motion from Lemma A.3. Hence

$$(I - \Pi_C) A_C = K_C, \qquad A_C^\top (I - \Pi_C) A_C = M_C^{-1}. \tag{55}$$

Let ζ_v denote the local meeting-point coordinates and $B_v = \partial X / \partial \zeta_v$. In the coordinate order (ζ_v, z_ℓ, z_r), the parent decoder Jacobian is

$$J_I = \begin{bmatrix} A_\ell B_v & J_\ell & 0 \\ -A_r B_v & 0 & J_r \end{bmatrix}. \tag{56}$$

The minus sign appears because the right child has endpoint displacement $D - X$.

Taking the Schur complement of the child-coordinate blocks in $J_I^\top J_I$ gives

$$S = B_v^\top \left[A_\ell^\top (I - \Pi_\ell) A_\ell + A_r^\top (I - \Pi_r) A_r \right] B_v \tag{57}$$

$$= B_v^\top \left(M_\ell^{-1} + M_r^{-1} \right) B_v. \tag{58}$$

The Schur complement removes motion within the two child fibers, leaving only the motion required to move their endpoints.

The block determinant identity now gives

$$\det(J_I^\top J_I) = \det(J_\ell^\top J_\ell) \det(J_r^\top J_r) \det\left[B_v^\top (M_\ell^{-1} + M_r^{-1}) B_v \right]. \tag{59}$$

Since B_v is square and nonsingular on a regular lens chart,

$$\det\left[B_v^\top (M_\ell^{-1} + M_r^{-1}) B_v \right] = \det(B_v)^2 \det(M_\ell^{-1} + M_r^{-1}).$$

Taking positive square roots yields

$$\mathcal{J}_I = \mathcal{J}_\ell \mathcal{J}_r |\det B_v| \sqrt{\det(M_\ell^{-1} + M_r^{-1})}. \tag{60}$$

Finally, the parent contains exactly the links in its two children, so

$$M_I = M_\ell + M_r. \tag{61}$$

Each merge therefore passes one scalar volume factor and one $d \times d$ matrix to its parent. With dense $d \times d$ linear algebra, a merge costs $O(d^3)$. A balanced tree has $O(N)$ nodes and $O(\log N)$ levels, giving $O(Nd^3)$ total work and $O(d^3 \log N)$ parallel depth. □

A.3.4 Terminal volume factors

The volume recursion starts from the two- and three-link leaves. The two-link factor follows directly from its coordinate map. Three links require a separate base case because the general recursion assumes full-rank endpoint maps for both children, which a single link does not have.

For a two-link leaf, let η be local coordinates for $v \in \mathbb{S}^{d-2} \subset D^\perp$ and write

$$G_v = (\partial_\eta v)^\top (\partial_\eta v).$$

Using the coordinates from above,

$$x_1 = \frac{D}{2} + hv, \qquad x_2 = \frac{D}{2} - hv, \qquad h = \sqrt{b^2 - \frac{\|D\|^2}{4}}.$$

At fixed D, differentiation with respect to η gives the pair Jacobian $(h\partial_\eta v, -h\partial_\eta v)$. Its Gram matrix is $2h^2 G_v$, so

$$\mathcal{J}_2 = (\sqrt{2}\, h)^{d-2} \sqrt{\det G_v}. \tag{62}$$

When $d = 2$, the chart has no continuous coordinates and this factor is 1 under the convention fixed at the start of the appendix.

For a three-link leaf, let $\eta \in \mathbb{R}^{d-1}$ parameterize the first link x_1, and write $A = \partial_\eta x_1$. The remaining two links form a two-link bridge with displacement $R = D - x_1$. Let J_2 be the Jacobian of that pair with respect to its own coordinates and let A_2 describe its response to a change in R. The full three-link Jacobian is

$$J_3 = \begin{bmatrix} A & 0 \\ -A_2 A & J_2 \end{bmatrix}. \tag{63}$$

The minus sign comes from $R = D - x_1$. Taking the Schur complement of $J_2^\top J_2$ and using Lemma A.3 gives

$$A^\top \left(I_d + M_2^{-1}\right) A.$$

The three-link volume factor is therefore

$$\mathcal{J}_3 = \mathcal{J}_2 \sqrt{\det\left[A^\top (I_d + M_2^{-1}) A\right]}. \tag{64}$$

Its endpoint-sensitivity matrix follows directly from the linkwise definition,

$$M_3 = M_2 + I_d - \widehat{x}_1 \widehat{x}_1^\top. \tag{65}$$

These two quantities initialize the recursive volume calculation at every regular terminal leaf.

A.3.5 Normalized bridge densities

The recursive volume factor turns a normalized coordinate density into a normalized density on the corresponding bridge chart.

Corollary A.4 (Normalized bridge density)**.** *Let U be an injective regular coordinate domain with normalized density q_U, and let $D_e : U \to \mathcal{B}_N(e)$ be its decoder. Then*

$$q_{\mathcal{B}}(x) = \frac{q_U(E_e(x))}{\mathcal{J}_e(E_e(x))}, \qquad x \in D_e(U), \tag{66}$$

with zero density outside $D_e(U)$.

Proof. Applying the change-of-variables formula from Section **??** gives

$$\int_{D_e(U)} q_{\mathcal{B}}(x)\, d\mu_{N,e}(x) = \int_U q_U(u)\, du = 1. \tag{67}$$

□

A.3.6 Sampling and feasible guidance

For sampling, draw $\varepsilon \sim p_0$, set $u = T_\psi(\varepsilon; c, e)$, and decode $x = D_e(u)$. The sample is feasible, and its intrinsic log density is

$$\log q_{\mathcal{B}}(x \mid c, e) = \log q_\psi(u \mid c, e) - \log \mathcal{J}_e(u). \tag{68}$$

The same decoder can be used during guidance. We optimize the sampler's base noise and decode before evaluating the objective, so every candidate scored during optimization remains in $\mathcal{B}_N(e)$. The optimizer and evaluation budget used for CDR-H3 design are given with the experimental details.

A.4 BOUNDED ALLOCATION COORDINATES

Recall that

$$\mathcal{A}_B = \left\{ x \in \mathbb{R}^n : \ell_i < x_i < u_i,\ \sum_{i=1}^{n} x_i = B \right\},$$

where $\sum_i \ell_i < B < \sum_i u_i$. Fix an ordered binary tree on the n components. For a subtree I, write $L_I = \sum_{i \in I} \ell_i$ and $U_I = \sum_{i \in I} u_i$ for its smallest and largest possible subtotals.

Proposition A.5 (Allocation coordinates and volume)**.** *The recursive split map*

$$D_B : (0,1)^{n-1} \to \mathcal{A}_B$$

is bijective. On every region where the active bounds are fixed, the map and its inverse are smooth, with intrinsic volume factor

$$\mathcal{J}_B(t) = \sqrt{n} \prod_{v \in \mathrm{Int}(T)} w_v(D_v), \qquad w_v(D_v) = \beta_v(D_v) - \alpha_v(D_v). \tag{69}$$

The same formula holds almost everywhere on $(0,1)^{n-1}$.

Proof. Consider a node with subtotal D and children ℓ and r. A left-child subtotal is feasible exactly when it satisfies its own bounds and leaves a feasible subtotal for the right child. Hence it must lie in

$$\alpha < D_\ell < \beta, \qquad \alpha = \max\{L_\ell, D - U_r\}, \qquad \beta = \min\{U_\ell, D - L_r\}. \tag{70}$$

Because the parent subtotal is strictly feasible, $L_\ell + L_r < D < U_\ell + U_r$, these bounds satisfy $\alpha < \beta$. We can therefore write

$$D_\ell = \alpha + t(\beta - \alpha), \qquad D_r = D - D_\ell, \qquad t \in (0,1).$$

Both child subtotals are again strictly feasible. Repeating the same split down the tree eventually assigns every leaf a value inside its bounds, and the child totals add back to B at the root.

The inverse follows from the same construction. Given $x \in \mathcal{A}_B$, sum the leaves below each node to recover its subtotal D_v. At an internal node, its left-child subtotal lies strictly between $\alpha_v(D_v)$ and $\beta_v(D_v)$, so

$$t_v = \frac{D_{v_\ell} - \alpha_v(D_v)}{w_v(D_v)} \in (0,1). \tag{71}$$

These coordinates are unique, and substituting them back into the decoder recovers the same child subtotals at every split. Thus D_B is bijective.

We next compute its volume factor. Write $F_B(t) = (x_1, \ldots, x_{n-1})$ for the first $n-1$ decoded components. The last component is fixed by the total, so the full allocation is

$$G_B(y) = \left(y_1, \ldots, y_{n-1}, B - \sum_{i=1}^{n-1} y_i \right), \qquad D_B = G_B \circ F_B.$$

The Gram matrix of G_B is $I_{n-1} + \mathbf{1}\mathbf{1}^\top$, whose determinant is n. Therefore

$$\mathcal{J}_B(t) = \sqrt{n}\, |\det DF_B(t)|. \tag{72}$$

To compute $\det DF_B$, let $a_v = D_{v_\ell}$ denote the left-child subtotal at each internal node v. Order the internal nodes from the root down the tree. The parent subtotal D_v depends only on coordinates at its ancestors, so

$$\frac{\partial a_v}{\partial t_v} = w_v(D_v),$$

and a_v does not depend on coordinates below v. The Jacobian from the split coordinates (t_v) to the subtotals (a_v) is therefore triangular, with determinant

$$\prod_{v \in \mathrm{Int}(T)} w_v(D_v).$$

These internal subtotals determine the first $n-1$ leaf values through a linear map whose determinant has absolute value one. This follows inductively from the tree: at each split, once the child subtotals are fixed, all but one leaf value in each child determine the remaining leaf by subtraction. The new parent subtotal enters that relation with coefficient one, while the two child transformations have determinant of absolute value one by induction. Hence

$$|\det DF_B(t)| = \prod_{v \in \mathrm{Int}(T)} w_v(D_v),$$

which gives the stated volume factor.

Finally, each α_v and β_v is the maximum or minimum of two affine functions of the parent subtotal. Once those choices are fixed, the decoder and inverse are smooth. A change of active bound occurs only where the two corresponding affine functions agree, giving lower-dimensional boundaries between these smooth regions. The volume formula therefore holds almost everywhere on the open coordinate domain. □

The same change-of-variables formula used for the bridge coordinates gives the normalized allocation density

$$q_{\mathcal{A}}(x \mid B) = \frac{q_T(E_B(x))}{\sqrt{n}\prod_v w_v(D_v)}. \tag{73}$$

The factor $\sqrt{n}$ comes from measuring volume in the sum hyperplane rather than in the first $n-1$ components. Once the subtree bounds are known, each tree node requires only constant scalar work. A balanced tree therefore gives $O(n)$ total work and $O(\log n)$ parallel depth.

B Experimental Details

Table 5 summarizes the main evaluation panels, sample counts, and resampling units. When several generated samples come from the same biological structure, we resample the parent PDB entry, CATH domain, or target family rather than the individual samples. For the synthetic experiments, we resample complete configurations, fibers, or targets as listed in the table.

Table 5: Summary of evaluation panels and budgets. Conditions and generated samples are reported separately from the units used to compute confidence intervals. For guided search, the budget is the number of decoded objective evaluations. A dash indicates that no search budget applies.

Experiment	Status	Independent units	Conditions / samples	Budget	Resampling unit
Bridge geometry and volume	Prospective synthetic	240 configurations	$N = 2, \ldots, 31$	-	Configuration
LoopGen preservation	Prospective	191 PDBs	2,061 / 20,610	10 samples	PDB
CATH preservation and closure	Confirmatory	256 domains	21,294 / 212,940	10 samples	Domain
FrameFlow preservation	Confirmatory	4 targets	73 / 292	4 samples	Target family
Synthetic RCC-coordinate guidance	Prospective synthetic	8 fibers	256 targets	64 calls	Target within fiber
CDR-H3 RCC-coordinate guidance	Prospective	255 PDBs	288 conditions	64 calls	PDB
Antibody search controls	Post hoc	255 PDBs	288 conditions	64 calls	PDB
Allocation correction	Prospective synthetic	8,192 loads	8,192 outputs	1 pass	Load
Allocation probability	Prospective synthetic	312 fibers	$n = 2, \ldots, 40$	-	Fiber

Before accessing the temporal test structures, we fixed the CDR-H3 checkpoint and full evaluation protocol, including the eligibility criteria, visible model inputs, design objective, random streams, candidate budget, bootstrap procedure, and prespecified decision criteria. The initial panel contained 292 conditions from 258 PDB entries. Four failed the prespecified structural eligibility criteria, leaving 288 eligible conditions from 255 PDBs. We performed the guide-then-repair, atom-aware, and alternative-search analyses after opening the panel and report them as post hoc throughout the paper.

C Numerical and Probabilistic Validation

C.1 Coordinate, volume, and gradient checks

We tested eight bridge configurations at each chain length $N = 2, \ldots, 31$. We split the eight configurations evenly between lower- and higher-extension endpoints, with $\|e\|/(Nb)$ ranging from 0.65 to

0.85. For each configuration, we computed the full decoder Jacobian by automatic differentiation. For prespecified subsets of these configurations, we also checked reduced-QR volume calculations, gradients, independent action-angle coordinates, and rotation/reflection consistency. For bounded allocations, we tested every component count from 2 through 40 and budget ratios from 0.035 through 0.965. Table 6 reports the largest discrepancy observed in each test.

Table 6: Numerical checks of the bridge and allocation coordinates. Entries show the maximum absolute discrepancy in float64. Volume checks compare the recursive volume factor with the full decoder Jacobian. Transition checks compare independently constructed overlapping charts. All tests are performed on regular chart domains.

Maximum absolute discrepancy	Protein bridges	Bounded allocations
Constraint residual	4.07×10^{-15}	3.55×10^{-15}
Encode-decode reconstruction	3.55×10^{-15}	2.28×10^{-15}
Base-noise inversion	1.78×10^{-14}	1.00×10^{-11}
Recursive versus full-Jacobian log volume	1.42×10^{-14}	2.84×10^{-14}
Coordinate-gradient agreement	1.91×10^{-14}	1.42×10^{-13}
Independent-chart area identity	1.42×10^{-14}	4.71×10^{-14}
Rotation/reflection or affine covariance	4.97×10^{-14}	1.03×10^{-12}

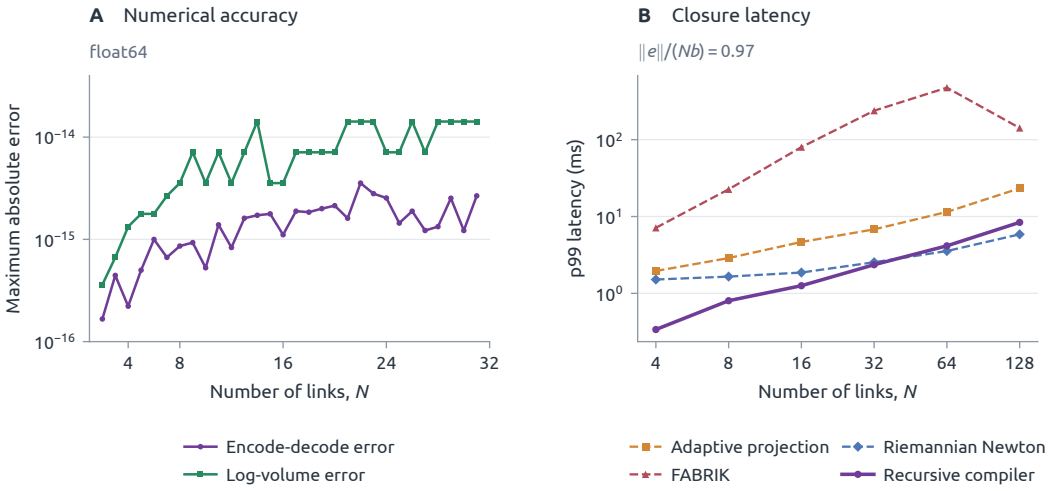


Figure 4: Numerical accuracy and closure latency across chain lengths. (A) Maximum encode-decode error and recursive-versus-Jacobian log-volume discrepancy across eight prespecified configurations at each N, computed in float64. (B) Post hoc 99th-percentile closure latency for batches of 128 float32 chains with $\|e\|/(Nb) = 0.97$. Logarithmic axes are marked by powers of ten. Timing results are specific to the measured implementation and hardware.

All tested configurations are included in these summaries. The smallest singular values of the bridge and allocation decoder Jacobians were 0.003821 and 1.51×10^{-4}, respectively. The derivative checks are restricted to regular chart pieces and therefore exclude collinear bridges, lens boundaries, allocation faces, and active-bound seams.

C.2 Conditional bridge density

We fit the conditional density on CATH-derived surrogate bridges with 8 and 16 links. Fit and holdout sets contained disjoint parent domains. For each bridge, the conditional, unconditional, and uniform-coordinate densities use the same intrinsic volume factor. Their likelihood differences therefore reflect differences in the fitted coordinate density rather than the choice of geometric measure.

All 2,040 inversions and likelihood evaluations were finite. We also drew 4,096 samples from the fitted models; every sample satisfied the bridge constraint, with a maximum endpoint residual of

Table 7: Conditional-density results on CATH-derived surrogate bridges. Log likelihood is measured relative to intrinsic Hausdorff volume and normalized by the bridge dimension. Differences are paired means across domains. Parentheses show 95% confidence intervals from 5,000 domain-level bootstrap resamples.

N	Examples	Conditional log p_H	Conditional − uniform	Conditional − unconditional	Standardized variance range
8	1,024	−1.1108	0.1017 (0.0885, 0.1141)	0.0140 (0.0106, 0.0173)	0.873-1.101
16	1,016	−1.1702	0.0763 (0.0682, 0.0844)	0.0048 (0.0029, 0.0066)	0.891-1.125

1.88×10^{-15}. The fitted densities are therefore normalized on the two surrogate bridge settings tested here. This experiment does not measure likelihood under the native protein distribution.

C.3 Common-target importance sampling and RCC-coordinate guidance

We compare three proposal distributions against the same Liouville action-angle target on eight prespecified synthetic fibers. Using a common target avoids the mismatch that arises when different samplers are evaluated under different target laws. Each proposal contributes 512 samples per fiber, and reference moments are computed from 4,096 independent samples drawn exactly from the target.

Table 8: Importance sampling under a shared action-angle target. Each proposal contributes $K = 512$ samples on each of eight synthetic fibers. Reference moments are computed from 4,096 independent target samples. ESS denotes effective sample size. Weight and moment-error columns report the maximum across fibers. Aggregate improvements need not hold on every fiber.

Proposal	Mean ESS/K ↑	Min ESS/K ↑	Max weight ↓	Mean error ↓	Log-var error ↓
Learned recursive proposal	**0.2496**	**0.0589**	**0.1642**	**0.1934**	**0.2250**
Uniform recursive coordinates	0.1273	0.0137	0.3520	0.3897	0.3741
CoBarS	0.1761	0.00202	0.9833	0.7286	3.6898

Two prespecified criteria were not met. The learned proposal's minimum ESS fraction was 0.0589, below the prespecified threshold of 0.08, and uniform recursive coordinates achieved higher ESS on one of the eight fibers. A 512-step Liouville MCMC control also failed the prespecified mixing diagnostic at 16 links. Increasing the chain to 4,096 steps reduced the maximum standardized mean and log-variance errors to 0.1682 and 0.0691, respectively, but the chain still had split-$\widehat{R} = 1.795$ and a minimum autocorrelation ESS fraction of 8.51×10^{-4}. These results do not show that the learned proposal is better on every fiber, and they do not support a general claim about MCMC mixing time.

We also test guidance through the reparameterized sampler while requiring every evaluated state to remain feasible. The experiment uses 256 targets across eight additional fibers. Guidance and IID sampling each receive 64 decoder and objective evaluations.

Table 9: Base-noise guidance on synthetic bridge targets. RMSD is averaged over 256 targets across eight fibers in normalized coordinates. IID sampling and guidance each receive 64 evaluations. Parentheses show paired 95% fiber-stratified bootstrap confidence intervals. Uniform-coordinate results were not part of the prespecified confirmation criteria.

Proposal	Best initial RMSD ↓	IID best@64 ↓	Guided best@64 ↓	Guided − IID (95% CI)	Guided − initial (95% CI)
Learned	1.50700	1.18050	0.88706	−0.29344 (−0.31983, −0.26718)	−0.61994 (−0.65305, −0.58765)
Uniform	1.54575	1.19862	0.89847	−0.30015 (−0.32878, −0.27160)	−0.64728 (−0.68822, −0.60888)

All 28,672 gradient evaluations were finite and nonzero. Across all targets, guided states, and IID samples, the largest link and endpoint residuals were 4.44×10^{-16} and 1.95×10^{-15}, respectively. Because the targets were drawn from the frozen learned proposal, this experiment tests guidance through the constrained sampler rather than biological design.

D Complete Protein-Generator Results

D.1 LoopGen preservation

We apply fixed-512 projection and RCC to the same 20,610 terminal states generated by three independently trained LoopGen models. Table 10 reports the pooled structural and timing results.

Table 10: Pooled LoopGen closure and preservation results. Both methods are applied to the same 20,610 samples from 2,061 condition-seed blocks across three trained models. RMSD is measured relative to the native loop, and diversity is the pairwise structural distance between generated samples. End-to-end and terminal-map timings are reported separately.

Metric	Fixed-512 projection	Recursive compiler
Best-of-10 native Cα RMSD (Å)	1.02253	1.02383
Best-of-10 native N/Cα/C RMSD (Å)	0.96653	0.96754
Sample-mean native Cα RMSD (Å)	1.70194	1.70184
Sample-mean native N/Cα/C RMSD (Å)	1.60141	1.60141
Pairwise Cα diversity (Å)	1.60510	1.60500
N/Cα/C nonlocal clash rate	0.01980	0.02009
End-to-end time (s)	25.883	24.694
Median terminal-map time (ms)	8.890	0.595

Table 11 shows the same comparison separately for each training seed. The structural differences remain small across all three models, while RCC is approximately $15\times$ faster at the terminal mapping step.

Table 11: LoopGen preservation across three training seeds. Differences are RCC minus fixed-512 projection, in Å. Map speedup is the ratio of fixed-512 to RCC terminal-map time. All three seeds satisfy the prespecified preservation criteria. Pooled differences use PDB-group weighting.

Seed	Δ best-10 Cα	Δ mean N/Cα/C	Δ diversity	Map speedup	Decision
20261101	$+0.00166$	-0.00169	-0.00055	$14.99\times$	Pass
20261102	$+0.00018$	$+0.00050$	$+0.00004$	$15.15\times$	Pass
20261103	$+0.00105$	$+0.00049$	-0.00017	$14.82\times$	Pass
Pooled	$+0.00096$	-0.00024	-0.00023	$14.99\times$	Pass

The PDB-resampled difference in best-of-10 Cα RMSD is $+0.00096$ Å ($+0.00010$, $+0.00180$). The corresponding differences are -0.00024 Å (-0.00080, $+0.00030$) for sample-mean N/Cα/C RMSD and -0.00023 Å (-0.00080, $+0.00030$) for diversity. We see the same preservation on the 96 conditions with loop lengths 10-13. For fixed-512 projection and RCC, respectively, best-of-10 Cα RMSD is 2.2519 and 2.2558 Å, sample-mean N/Cα/C RMSD is 3.2569 and 3.2556 Å, and diversity is 3.1842 and 3.1831 Å.

D.2 CATH fragment preservation

We evaluate all 21,294 conditions from 256 reserved domains in the CATH 4.4.0 S40 nonredundant set (Sillitoe et al., 2021). This yields 212,940 paired samples for each closure method, with no failures. The confidence intervals in the main text resample domains, whereas Table 12 reports condition-weighted aggregates over the full panel.

Relative to fixed-512 projection, recursive compilation changes best-of-10 RMSD by $+0.002877$ Å ($+0.002260$, $+0.003487$), sample-mean RMSD by $+0.007947$ Å ($+0.007480$, $+0.008416$), diversity by $+0.014108$ Å ($+0.013366$, $+0.014841$), and clash rate by -0.000401 (-0.000803, $+0.000001$). Every prespecified length and endpoint-extension stratum remains within its preservation margin, including the 13 conditions with endpoint fraction at least 0.9.

Structural examples. Figure 5 shows two deterministically selected examples from the same confirmation panel. The representative condition has the compiler-minus-adaptive RMSD effect

Table 12: Condition-weighted CATH fragment-preservation results. All methods are applied to the same 212,940 paired samples from 21,294 conditions in 256 domains. RMSD and pseudo-angle error are measured relative to the native fragments. MAE denotes mean absolute error. These summaries are condition-weighted and therefore differ from the domain-weighted paired comparisons used for uncertainty estimation in the main text.

Metric	Fixed-512 projection	Adaptive projection	Recursive compiler
Best-of-10 Cα RMSD (Å)	6.34301	6.34301	6.34586
Sample-mean Cα RMSD (Å)	10.02369	10.02368	10.03165
Pairwise Cα diversity (Å)	9.47354	9.47354	9.48769
Pseudo-angle MAE (degrees)	22.30784	22.30778	22.32319
Nonlocal Cα clash rate	0.328205	0.328205	0.327792
Maximum normalized link residual	5.96×10^{-8}	5.01×10^{-5}	5.96×10^{-8}
Map time (s)	18.273	1.144	1.002
End-to-end time (s)	214.905	197.777	197.634

closest to the panel median, and its displayed sample has the within-condition displacement closest to the median. The second condition has the largest endpoint fraction in the panel, with its sample chosen by the same rule. Each row uses the same generator output and camera.

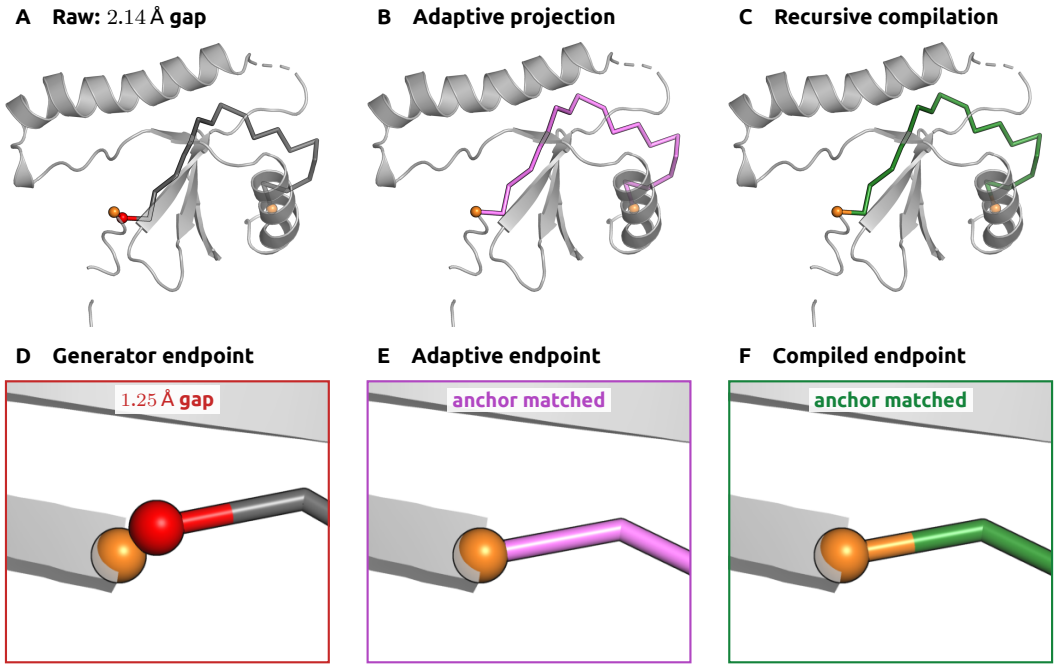


Figure 5: CATH closure at representative and high-extension conditions. (A-C) Generator output, adaptive projection, and recursive compilation for the condition closest to the median paired effect. (D-F) The same comparison for the condition with the largest endpoint fraction, shown with a magnified view near the terminal anchor. Magenta denotes adaptive projection, green denotes recursive compilation, and orange marks the fixed anchors. Each row uses the same input and camera. The examples illustrate endpoint closure and local structural differences, not atomistic quality or biological designability.

D.3 FrameFlow motif-linker preservation

We evaluate 292 FrameFlow samples from 73 conditions across four held-out target families. Fixed-512 projection and RCC are applied to the same raw linkers, while the fixed motifs remain unchanged.

Table 13: FrameFlow motif-linker geometry before and after closure. The panel contains 292 samples from 73 conditions across four held-out target families. Fixed-512 projection and RCC are applied to the same raw linkers and leave the fixed motifs unchanged. CA-ProteinMPNN negative log likelihood (NLL) is used as a measure of whole-backbone sequence compatibility rather than folding or binding.

Metric	Raw FrameFlow	Fixed-512 projection	Recursive compiler
RMS displacement from raw (Å)	0	0.14269	0.14321
Pairwise Cα diversity (Å)	7.38853	7.31771	7.31749
Radius of gyration (Å)	6.72415	6.69332	6.69163
Pseudo-angle change (degrees)	0	0.44578	0.47024
Nonlocal Cα clash rate	0.0001547	0.0001398	0.0001398
CA-ProteinMPNN NLL (nats/residue)	1.22926	1.22871	1.22816
Pairwise sequence diversity	0.35661	0.35777	0.36160
Maximum link residual (Å)	1.56908	2.38×10^{-6}	2.38×10^{-6}

The RCC-minus-fixed difference in RMS displacement is $+0.001720$ Å (-0.006441, $+0.007899$) after resampling target families. The corresponding differences are $+0.002379$ Å (-0.003360, $+0.010869$) for pairwise structural diversity, -0.000413 (-0.001968, $+0.001028$) for CA-ProteinMPNN NLL, and $+0.001466$ (-0.006209, $+0.007598$) for sequence diversity. The two closure methods have the same nonlocal clash rate.

D.4 Full-backbone reconstruction and restrained relaxation

We use PULCHRA (Rotkiewicz & Skolnick, 2008) to reconstruct N/Cα/C/O atoms for four fixed conditions at each CATH length from 5 through 25, with four paired samples per condition. All 1,764 reconstructions complete successfully. We also reconstruct the native Cα traces to measure the error introduced by PULCHRA itself, independently of the generator or closure method.

Table 14: PULCHRA reconstruction on 84 CATH conditions. Reconstructed N/Cα/C/O backbones are compared with the corresponding native structures. MAE denotes mean absolute bond-angle error. Reconstruction from native Cα traces serves as a control for error introduced by PULCHRA. Lower RMSD, angle error, and clash rate are better.

Input	Backbone RMSD (Å)	Angle MAE (degrees)	Clash rate
Native Cα oracle	0.3919	4.1338	0.0476
Adaptive projection	9.9375	7.8436	0.3810
FABRIK	9.9508	7.9276	0.4018
Riemannian Newton	9.9384	7.8592	0.3839
Recursive compiler	9.9418	7.8233	0.3780

Relative to adaptive projection, RCC changes backbone RMSD by $+0.00410$ Å (-0.00528, $+0.01366$), bond-angle MAE by -0.0179 degrees (-0.0999, $+0.0632$), and clash rate by -0.00389 (-0.01889, $+0.01111$). We then apply restrained OpenMM relaxation to 504 reconstructed backbones, all of which complete successfully. After relaxation, the RCC-minus-adaptive difference in backbone RMSD is -0.01249 Å (-0.03276, $+0.00650$), and both methods have zero nonlocal heavy-atom clashes.

D.5 Closure convergence and scaling

The eight closure methods in Table 2 are evaluated on the same 212,940 CATH terminal states using a common feasibility tolerance of 5×10^{-5}. Table 15 reports iteration counts for the iterative methods where they were recorded. Unrecorded iteration counts are reported as n/r.

We also measure closure latency in a controlled scaling experiment with 128 float32 chains for each combination of $N \in \{4, 8, 16, 32, 64, 128\}$ and endpoint fraction $\{0.10, 0.50, 0.90, 0.97\}$. Each case is timed over 15 repeats using eight CPU threads. All 12,288 outputs across methods and test cases satisfy the same feasibility tolerance.

Table 15: Closure iterations on 212,940 shared CATH terminal states. Columns report iteration quantiles, the maximum recorded iteration count, and the largest residual. All methods satisfy the common feasibility tolerance. n/r indicates that the corresponding iteration count was not recorded. FCCD-1, consensus ADMM, and CoBarS were added post hoc.

Closure method	p50	p95	p99	Maximum	Max residual
Fixed-512 projection	512	512	512	512	1.94×10^{-6}
Adaptive projection	16	n/r	512	512	5.01×10^{-5}
FABRIK	1	n/r	2	n/r	2.53×10^{-5}
FCCD-1	6	18	34	248	2.58×10^{-5}
Riemannian Newton	2	n/r	3	n/r	2.55×10^{-5}
Consensus ADMM	16	22	27	73	2.59×10^{-5}
CoBarS	3	5	7	14	1.91×10^{-6}
Recursive compiler	1	1	1	1	1.94×10^{-6}

Table 16: Closure latency across chain lengths and endpoint extensions. Each case contains 128 float32 chains and is timed over 15 repeats using eight CPU threads. We report the median of the case-level p50 latencies and the largest case-level p99 latency, in milliseconds.

Closure method	Median case p50	Largest case p99
Adaptive projection	2.059	23.29
FABRIK	3.588	475.61
Riemannian Newton	1.449	5.86
Recursive compiler	1.395	17.51

FABRIK has the lowest p99 latency on the observed CATH samples, but develops a long latency tail as the endpoint approaches maximum extension in the scaling experiment. RCC uses a fixed sequence of recursive operations, but is not uniformly faster in wall-clock time. Its computational work is predictable rather than minimal in every case.

D.6 Guidance and search through a frozen generator

We use two held-out studies to test guidance through a frozen protein generator. The first compares gradients through RCC with a stop-gradient control using the same forward map. The second compares fixed-budget search methods while applying a closure map before every score. These studies use separate held-out panels and are distinct from the prospective CDR-H3 experiment.

Table 17: Latent guidance through a frozen generator and closure map. RCC and the stop-gradient control produce identical outputs before guidance, so their difference isolates the effect of gradients through the compiler. Hard-clash counts are shown before and after search. Link and endpoint errors are measured in normalized coordinates, and cycle counts refer to the closure operation.

Constraint map	Objective reduction ↑	Hard clashes ↓	Max link error ↓	Max endpoint error ↓	Cycles mean/max
Adaptive projection	45.38%	$147 \to 27$	4.95×10^{-5}	5.57×10^{-7}	311.95/512
FABRIK	46.21%	$172 \to 33$	5.96×10^{-8}	4.48×10^{-6}	1.00/1
Recursive compiler	45.87%	$146 \to 29$	5.96×10^{-8}	5.12×10^{-7}	1.00/1 fixed
Stop-gradient compiler	42.17%	$146 \to 28$	5.96×10^{-8}	5.87×10^{-7}	1.00/1 fixed
Unconstrained	47.84%	$197 \to 28$	4.40×10^{-1}	2.20	0/0

The paired recursive-gradient minus stop-gradient difference in objective change is -0.001900 (-0.002302, -0.001535), showing a benefit from differentiating through the compiler. The prespecified criterion also required the recursive-gradient arm not to increase hard clashes. It ends with 29 clashes, compared with 28 for the stop-gradient control, so this criterion is not met. The one-clash difference is statistically unresolved.

We next compare fixed-budget search methods using the same frozen generator. Every candidate is passed through its assigned closure map before the objective is evaluated.

Table 18: Batched latent search through a frozen generator. Each row evaluates 46,336 candidates. Hard-clash counts are shown before and after search, and endpoint errors are measured in normalized coordinates. Closure work counts compiler passes or solver sweeps rather than floating-point operations. CMA-ES (Hansen, 2016) was added post hoc; the remaining search comparison was specified prospectively.

Search	Closure	Objective reduction ↑	Hard clashes ↓	Closure work ↓	Max endpoint error ↓	Time (s)
Random	Recursive compiler	49.91%	209 → 17	46,336	6.88×10^{-7}	20.32
CEM*	Recursive compiler	57.96%	209 → 13	46,336	1.36×10^{-6}	28.98
CMA-ES†	Recursive compiler	58.68%	209 → 13	46,336	1.36×10^{-6}	37.17
Hierarchical tree search	Recursive compiler	54.75%	209 → 12	46,336	9.54×10^{-7}	29.18
Hierarchical tree search	Adaptive projection	54.57%	200 → 11	16,484,796	1.05×10^{-6}	28.76
Hierarchical tree search	FABRIK	52.45%	214 → 16	47,053	2.38×10^{-5}	28.69

*Prospectively selected search. †Added after the panel was opened but before its own outcome was observed. End-to-end timings are descriptive because the generator and objective dominate the total runtime.

CEM improves on random search, with a condition-bootstrap interval of -0.005703 to -0.003718 for the paired objective-change difference. The CMA-ES-minus-CEM difference is -0.000419 (-0.001137, $+0.000311$), so the two methods are not resolved on this panel. Adaptive projection requires 355.8 times as many closure sweeps as RCC, while FABRIK requires 1.015 times as many. RCC therefore makes the amount of closure work predictable, but this does not translate into a consistent end-to-end speedup when generator and objective evaluation dominate the runtime.

E Complete Prospective Antibody Results

E.1 Prespecified decision criteria

Table 19 reports the decision criteria fixed before we accessed the CHIMERA-Bench temporal CDR-H3 panel (Ahmed et al., 2026). All objective terms use only the visible antibody framework and epitope context. Native H3 coordinates and sequence are not used by the density model, design objective, candidate ranking, or success metric.

Table 19: Prespecified decision criteria for the prospective CDR-H3 study. The initial panel contains 292 conditions, including four that fail the structural eligibility criteria. Structural results are evaluated on the remaining 288 conditions. Native H3 coordinates and sequence are not used by the density model, design objective, ranking, or success criteria. These criteria evaluate the computational design procedure and do not establish biological function.

Criterion	Observed	Decision
Account for 292 conditions; at least 285 eligible	292; 288 eligible	Pass
Mean learned minus uniform density is positive	$+9.0739$ nats	Pass
True context beats permuted context in mean and frequency	$+1.1522$; 0.613	Pass
Objective CI for RCC-coordinate guidance minus IID lies below zero	-0.10083 to -0.07931	Pass
Joint success improves by at least 0.10	$0.7014 - 0.4201 = 0.2813$	Pass
Cα clash-free rate is noninferior	0.9861 vs. 0.8160	Pass
Diversity is at least 80% of learned IID	$5.841/5.343 = 1.093$	Pass
Every guidance gradient is finite and nonzero	17,280 / 17,280	Pass
Every link and endpoint residual is at most 5×10^{-5}	2.38×10^{-7}; 8.18×10^{-7}	Pass

The learned proposal assigns higher density than uniform recursive coordinates in 92.0% of conditions. RCC-coordinate guidance improves the design objective over learned IID sampling on 278 of the 288 eligible conditions. All 18,432 decoded candidates are included in the analysis, and rerunning the selected candidates reproduces every segment bitwise.

Figure 3 shows condition `8yvn_E_I_D`, selected after the study by a deterministic rule among successful examples with H3 lengths 12-16. It is shown only for visualization and does not enter the quantitative comparisons.

Algorithm 2 gives the guidance procedure used for the prospective CDR-H3 study. Each candidate is decoded before the objective is evaluated, so every state considered during optimization satisfies the bridge constraints.

Algorithm 2 Feasible guidance in RCC coordinates

Require: context c, endpoint e, base law p_0, reparameterization T_ψ, loss L, candidates K, and update steps S
Ensure: best bridge $x^\star \in \mathcal{B}_N(e)$

draw $\varepsilon^{(1)}, \dots, \varepsilon^{(K)} \sim p_0$
$(x^\star, L^\star) \leftarrow (\perp, +\infty)$
for $t = 0, \dots, S$ **do**
  for all $k = 1, \dots, K$ **do**
    $u^{(k)} \leftarrow T_\psi(\varepsilon^{(k)}; c, e)$
    $x^{(k)} \leftarrow \textsc{DecodeBridge}(e, N, u^{(k)})$
    $L^{(k)} \leftarrow L(x^{(k)}; c)$
  $k^\star \leftarrow \arg\min_k L^{(k)}$
  if $L^{(k^\star)} < L^\star$ **then**
    $(x^\star, L^\star) \leftarrow (x^{(k^\star)}, L^{(k^\star)})$
  if $t < S$ **then**
    update each $\varepsilon^{(k)}$ by one Adam step using $\nabla_{\varepsilon^{(k)}} L(x^{(k)}; c)$
return $x^\star$

For the prospective study, we use $K = 4$ initial candidates and $S = 15$ update steps. Each candidate is scored once before optimization and once after each update, giving $4 \times 16 = 64$ decoded objective evaluations per condition. Every evaluation is therefore performed on a backbone that satisfies the endpoint and link-length constraints.

E.2 Cartesian guide-then-repair

We compare RCC-coordinate guidance with Cartesian optimization followed by a terminal closure step. Both methods start from the same four learned samples and use the same 15-step search with 64 objective evaluations per condition. In the Cartesian arm, each link vector is optimized independently, and the resulting candidates are then closed with one of four terminal methods. Because backward-pass and closure costs differ, the comparison matches objective evaluations rather than total computation.

Table 20: Post-hoc Cartesian guidance followed by terminal closure. All 288 conditions use the same initial samples and objective-evaluation budget as RCC-coordinate guidance. Objective differences are reported relative to direct guidance, with 95% confidence intervals obtained by resampling the 255 PDB groups. Diversity is measured in Å. The raw Cartesian candidates do not satisfy the bridge constraints.

Terminal map	Objective ↓	Success ↑	Cα clean ↑	Diversity	Objective difference (95% CI)
Raw Cartesian guide (infeasible)	0.00755	0.0000	1.0000	6.906	−0.0090 (−0.0139, −0.0056)
Adaptive projection	0.64968	0.0764	0.4861	6.310	+0.6332 (+0.5547, +0.6889)
FABRIK	0.67007	0.0660	0.4549	7.784	+0.6535 (+0.5710, +0.7039)
CoBarS	0.69009	0.0694	0.4479	6.524	+0.6736 (+0.5951, +0.7338)
Recursive retraction	0.54716	0.1111	0.5347	6.170	+0.5306 (+0.4589, +0.5744)
RCC-coordinate guidance	0.01653	0.7014	0.9861	5.841	Reference

Before closure, the Cartesian candidates have a maximum normalized link error of 3.4078. All four terminal methods successfully close all 1,152 final candidates. Adaptive projection takes 16 iterations at the median and 512 at p95, while FABRIK takes one sweep at both quantiles. CoBarS takes three iterations at the median and five at p95, and the RCC retraction uses a single fixed pass. Every terminal method produces feasible outputs, so the lower success relative to RCC-coordinate guidance is not due to failed closure.

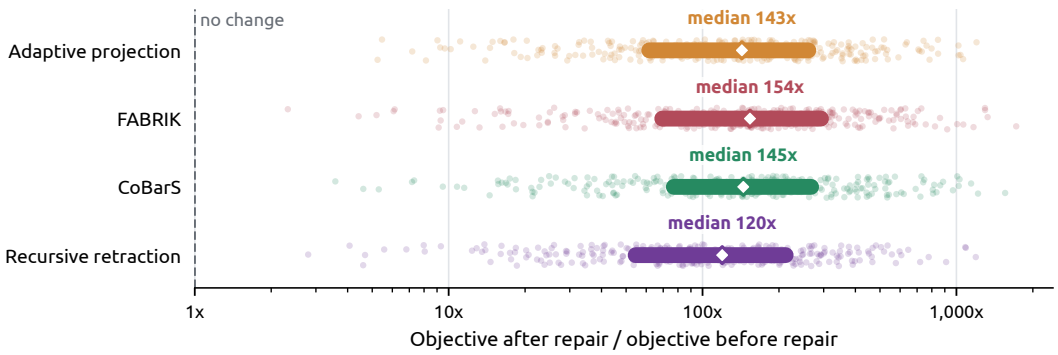


Figure 6: Objective change after terminal repair of Cartesian-guided designs. Each point shows the ratio of the objective after repair to the objective before repair for one condition. Because the objective is minimized, values above one indicate worse performance after repair. Thick bars show the interquartile range and diamonds mark the median. All 1,152 repaired outputs are feasible, yet every ratio is greater than one, with median increases of $120\times$-$154\times$. This analysis is post hoc.

E.3 Alternative search policies

We compare several search methods while keeping the RCC representation fixed. CEM, CMA-ES, UCT, and score-guided Langevin all search the same RCC coordinates, with 64 decoded objective evaluations per condition in the primary comparison. The differences therefore reflect the search policy rather than the constraint representation.

Table 21: Alternative search policies relative to Adam coordinate guidance. Differences are each post-hoc policy minus the prospective Adam guide at 64 objective evaluations per condition. Parentheses show 95% confidence intervals from 10,000 resamples of the 255 PDB-group means. Negative objective differences and positive success differences favor the alternative policy.

Search policy	Objective difference	Success difference
CEM	$+0.03816$ ($+0.03320$, $+0.04333$)	-0.2876 (-0.3575, -0.2176)
CMA-ES	$+0.04449$ ($+0.03705$, $+0.05429$)	-0.2784 (-0.3510, -0.2046)
UCT chart tree	$+0.21825$ ($+0.18584$, $+0.25248$)	-0.3490 (-0.4203, -0.2791)
Score-guided Langevin	$+0.00223$ (-0.00294, $+0.00606$)	$+0.0033$ (-0.0725, $+0.0810$)

At the primary 64-evaluation budget, Adam has lower objective values and higher success than CEM, CMA-ES, and UCT. Its differences from score-guided Langevin are unresolved on both measures. Langevin reaches a success rate of 72.2%, compared with 70.1% for Adam, but produces lower top-four diversity, 3.143 versus 5.841 Å.

We also repeat the comparison over budgets from 8 to 128 decoded objective evaluations per condition.

Table 22: Post-hoc CDR-H3 search across candidate budgets. Each cell reports joint structural success followed by mean objective over 288 conditions. All policies search RCC coordinates and receive B decoded objective evaluations per condition. The budgets match objective evaluations rather than backward computation or elapsed time.

B	Learned IID	CEM	CMA-ES	UCT	Adam guidance	Score Langevin
8	.188 / .2815	.097 / .3121	.201 / .2693	.149 / .3600	.323 / .2388	.267 / .1696
16	.219 / .2059	.160 / .2162	.264 / .1980	.285 / .2805	.531 / .1025	.490 / .0770
32	.330 / .1547	.271 / .1139	.406 / .1060	.292 / .2613	.635 / .0412	.622 / .0347
64	.417 / .1163	.427 / .0546	.441 / .0603	.354 / .2342	.701 / .0165	.722 / .0185
128	.510 / .0808	.566 / .0234	.424 / .0317	.358 / .2377	.733 / .0047	.764 / .0134

Across the budget sweep, Adam has higher success than every gradient-free method at each budget, with all paired PDB-group confidence intervals excluding zero. At 16 evaluations, Adam reaches 53.1% success, above the best 64-evaluation gradient-free result of 44.1%. Score-guided Langevin follows a similar trend, and its success difference from Adam remains unresolved at all five budgets. Every decoded candidate satisfies the bridge constraints. At the primary 64-evaluation budget, native Cα RMSD increases from 5.56 to 6.01 Å. This experiment measures context-directed design rather than reconstruction of the crystallographic H3 loop.

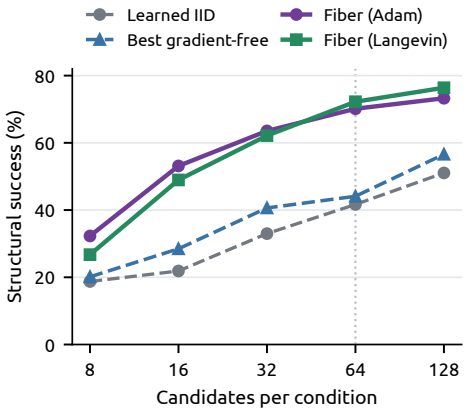


Figure 7: Structural success across candidate budgets. Post-hoc search on 288 conditions. All policies use RCC coordinates. The dashed gradient-free curve shows the best success rate among CEM, CMA-ES, and UCT at each budget. The vertical line marks the primary 64-evaluation comparison.

E.4 Atomistic and sequence checks

We reconstruct the native, uniform, learned-IID, and prospectively guided Cα traces with PULCHRA. All 1,152 reconstructions complete successfully, and the largest shift in a reconstructed Cα position is 0.00085 Å. The prospective guidance objective does not include atom-level clearance from the epitope. We therefore ran a post-hoc atom-aware variant that adds differentiable clearance terms against the visible epitope and framework atoms while keeping the same 64-evaluation search budget.

Table 23: Atomistic checks of CDR-H3 inference policies. Learned IID and the original coordinate-guidance outputs were generated prospectively. PULCHRA evaluation and the atom-aware guidance experiment were performed post hoc. Clean fractions denote the absence of clashes under the corresponding Cα or reconstructed atom-level criterion. These checks measure steric geometry rather than folding or binding.

Inference policy	Objective ↓	Success ↑	Cα clean ↑	PULCHRA clean ↑	Diversity (Å)
Learned IID	0.1070	0.4201	0.8160	0.2326	5.343
Prospective RCC-coordinate guidance	0.0165	0.7014	0.9861	0.1979	5.841
Post-hoc atom-aware RCC-coordinate guidance	0.0555	0.6528	1.0000	0.7500	6.454

In this post-hoc analysis, atom-aware guidance increases structural success relative to learned IID sampling, with a PDB-resampled 95% confidence interval of $+0.1562$ to $+0.3183$, and reduces the probability of any PULCHRA steric clash, with an interval of -0.5804 to -0.4392.

We also evaluate sequence compatibility on a separate set of 64 PDBs using ProteinMPNN. Atom-aware guidance has a mean NLL of 0.2262 nats per residue, compared with 0.1856 for learned IID, a difference of $+0.0406$ ($+0.0017$, $+0.0814$). Its normalized Hamming diversity is 0.1733, compared with 0.1528 for IID; the difference interval -0.0163 to $+0.0574$ includes zero. These checks do not establish folding, expression, affinity, specificity, or sequence designability.

E.5 Feasibility-maintaining guidance controls

We compare RCC with four additional methods that remain feasible throughout optimization on the development partition. These are projected Adam with FABRIK, normalized Riemannian gradient with Newton closure, Adam differentiated through Newton closure, and Adam differentiated through the RCC retraction. The panel contains 203 conditions from 199 PDBs. All methods begin from the same four learned proposals and optimize the same atom-aware objective, with no penalty for moving

away from the initial latent variables. Each run uses 15 updates followed by a final evaluation, for a total of 64 feasible objective evaluations. The IID baseline includes the same four initial proposals among its 64 samples. For every method, we retain the best objective value encountered during search. Neither guidance nor selection uses native H3 coordinates or sequence.

For the Adam-based methods, we test step sizes of 0.03, 0.10, and 0.30. For normalized Riemannian gradient, we test 0.10, 0.30, and 1.00. Three search seeds give 9,744 runs in total, all of which complete successfully. We select one setting per method by first preferring runs without failures and then maximizing the fraction that satisfy both Cα structural success and PULCHRA steric cleanliness. Steric cleanliness, objective value, and step size are used in that order to break ties. Because these settings are selected on the development panel, the comparison is descriptive rather than an independent confirmation. The combined criterion measures structural and steric quality and does not establish folding, binding, or developability.

Table 24: Feasibility-maintaining guidance on the development panel. Each selected row contains 609 condition-search-seed runs from one training checkpoint. Every scored state satisfies the bridge constraints. "Combined" requires both Cα structural success and PULCHRA steric cleanliness. Mean CPU times include initialization and search but not reconstruction. Methods receive the same number of objective evaluations rather than the same wall-clock time.

Method	Step	Objective ↓	Combined ↑	PULCHRA clean ↑	Cα success ↑	Mean ms ↓
RCC-coordinate Adam	0.10	0.31449	47.29%	72.91%	66.01%	52.65
Riemannian gradient + Newton	0.30	0.12300	46.80%	68.14%	67.32%	21.07
Adam through Newton	0.10	0.13652	41.22%	77.01%	54.02%	26.32
Adam through RCC retraction	0.10	0.14523	44.33%	77.34%	54.68%	63.90
Projected Adam + FABRIK	0.03	0.39226	30.05%	62.07%	44.99%	26.67
Learned IID	-	1.61614	8.70%	44.33%	22.66%	8.11

After selecting these settings, crossed resampling over search seeds and PDBs gives an RCC-minus-Riemannian difference of $+0.53$ percentage points in the combined criterion, with a descriptive 95% interval of -6.39 to $+7.37$. The corresponding differences from Adam through Newton and Adam through the RCC retraction are also unresolved: $+5.92$ (-1.96, $+13.57$) and $+3.07$ (-4.52, $+10.55$) points. RCC exceeds projected Adam with FABRIK by $+17.50$ points ($+9.66$, $+25.24$) and learned IID by $+38.47$ points ($+31.29$, $+45.48$). The comparison therefore shows improvement over IID sampling and this projected-Adam baseline, but not an advantage over the stronger feasibility-maintaining controls. For context, reconstruction of the native Cα traces is PULCHRA-clean in 176 of 203 cases.

The timings in Table 24 use a batched geometry-only implementation of the same RCC charts. Because the guidance objective does not use density, this implementation omits the intrinsic-volume calculation performed by the original decoder. Repeating the selected settings with the faster implementation preserves all 3,654 success outcomes, with a maximum change of 3.02×10^{-7} normalized units in the selected segments across methods. For RCC, the geometry-only implementation is $2.75\times$ faster than the original decoder, but its mean runtime is still $2.50\times$ that of Riemannian guidance. These sequential CPU timings are implementation-specific and should not be interpreted as an equal-wall-time comparison of search quality.

Training-seed replication. After fixing the settings above, we train two additional 106,850-parameter proposal models using the same 5,000-step training procedure and changing only the training seed. These replicas do not affect checkpoint or step-size selection. We evaluate each model on the same 203 development conditions, three search seeds, and six guidance methods. Across the original checkpoint and the two replicas, all 10,962 runs complete successfully. Because every checkpoint is evaluated on the same development panel, these runs provide replication across training seeds rather than additional independent test examples.

Resampling across training seeds, search seeds, and PDBs gives an RCC-minus-Riemannian difference of -0.95 percentage points, with a descriptive 95% interval of -7.31 to $+5.39$. The differences from Adam through Newton and Adam through the RCC retraction are also unresolved. RCC exceeds learned IID by $+35.62$ points ($+29.59$, $+41.58$). The replication therefore preserves the improvement over IID sampling but does not show an advantage over the strongest feasibility-maintaining

Table 25: Selected guidance settings across three training checkpoints. Entries show the percentage of runs satisfying both Cα structural success and PULCHRA steric cleanliness, with 609 runs per method and checkpoint. Settings were selected using the original checkpoint and then held fixed for both replicas. All three checkpoints are evaluated on the same development panel.

Method	Original seed	Replica 1	Replica 2	Mean
RCC-coordinate Adam	47.29	40.39	44.17	43.95
Riemannian gradient + Newton	46.80	44.01	44.33	45.05
Adam through Newton	41.22	42.04	44.50	42.58
Adam through RCC retraction	44.33	43.19	43.51	43.68
Projected Adam + FABRIK	30.05	24.47	28.90	27.81
Learned IID	8.70	8.37	8.05	8.37

controls. With only three training seeds, it provides limited information about variation across model training runs.

E.6 Independent sequence-design and refolding pilot

We run a separate development pilot on 32 PDBs selected by identifier hash, without using model outcomes, with one condition per PDB. Each of the six guidance policies uses its previously selected setting and the first fixed guidance seed. For every resulting backbone, we draw two paired ProteinMPNN sequence samples.

We first reconstruct each backbone with PULCHRA and then redesign only the H3 sequence with ProteinMPNN (Dauparas et al., 2022), masking the native H3 identities. We refold the resulting heavy- and light-chain variable domains with a separately trained, sequence-only ABodyBuilder3 model (Kenlay et al., 2024; Abanades et al., 2023), without providing antigen structure. Framework alignment excludes all CDRs, and self-consistency is measured by the Cα RMSD between the predicted and generated H3 loops. Neither ProteinMPNN nor ABodyBuilder3 is used to select the original backbone proposal or tune the guidance procedure.

Table 26: Sequence redesign and independent refolding on 32 development PDBs. All 480 sequence evaluations complete successfully. Means are computed over the listed sequence attempts; two sequence samples from the same PDB do not constitute independent PDB replicates. The 2 Å threshold is a fixed descriptive self-consistency criterion rather than experimental evidence of folding. Native controls are included to characterize the sequence-design and refolding pipeline.

Input backbone / policy	Attempts	Mean H3 Cα RMSD (Å)	RMSD $\leq$ 2 Å	Mean H3 pLDDT
RCC-coordinate Adam	64	6.79	0/64	73.22
Riemannian gradient + Newton	64	6.85	0/64	72.33
Adam through Newton	64	7.64	0/64	73.23
Adam through RCC retraction	64	7.63	0/64	72.57
Projected Adam + FABRIK	64	7.34	0/64	72.65
Learned IID	64	6.95	1/64	73.26
Native backbone, H3 redesign	64	4.28	8/64	74.94
Native backbone and sequence	32	2.55	17/32	79.42

None of the guided methods produces a refolded H3 within 2 Å of the generated backbone in this pilot. These counts remain zero when we additionally require at least one generated antigen contact and contact F1 of at least 0.5. Under this stricter criterion, the native-sequence control decreases from 17/32 to 16/32. Because several paired comparisons contain no successes in either arm, their bootstrap distributions are degenerate and do not provide a meaningful estimate of uncertainty.

The native controls also show that the sequence-design and refolding pipeline is imperfect even when starting from native structures. We have not established whether the public development PDBs overlap with the training data used by the refolding model. This pilot provides no evidence that the guided backbones are sequence-structure self-consistent under this independent refolding test. It

also shows that exact geometric closure and low reconstructed steric clash rates are not sufficient, by themselves, to demonstrate biological design quality.

E.7 Exact closure with minimal vertex displacement

We use eight PDBs from the development panel to study how terminal closure changes a frozen LoopGen output. For each PDB, we consider two ways of placing the generated loop before closure. The first uses the midpoint between the anchors. The second predicts a placement from same-length training loops using only the observed anchors and framework tangents, with the query PDB excluded from the donor set. Neither placement changes the LoopGen model or its generated backbone.

For each backbone, we express the closed structure in RCC coordinates and optimize the positions of the hidden Cα vertices while preserving the fixed endpoints and link lengths exactly. The objective uses only displacement from the generated Cα trace and does not use native loop coordinates, sequence, antigen context, or a refolding score. The 500-step optimization setting was chosen on these same development PDBs, so the results are descriptive rather than an independent evaluation.

Table 27: Closure distortion and refolding on frozen LoopGen outputs. Each row contains 32 sequence attempts from eight development PDBs, two generated backbones per PDB, and two paired ProteinMPNN sequence samples per backbone. Cα displacement is measured from the raw generator trace before atom reconstruction. Max link error is averaged over the per-backbone maxima after reconstruction. No generated backbone satisfies the 2 Å self-consistency threshold.

Placement	Closure	Cα displacement (Å)	Max link error (Å)	Refolding RMSD (Å)	RMSD $\leq$ 2 Å
Midpoint	None	0.00	6.8751	14.32	0/32
Midpoint	One-shot RCC	6.15	0.0008	12.10	0/32
Midpoint	One-shot Riemannian	6.43	0.0008	12.03	0/32
Midpoint	RCC chart + displacement minimization	**2.41**	0.0008	13.81	0/32
Training center	None	0.00	10.8455	8.74	0/32
Training center	One-shot RCC	9.87	0.0008	13.42	0/32
Training center	One-shot Riemannian	9.99	0.0009	13.04	0/32
Training center	RCC chart + displacement minimization	**4.10**	0.0008	**7.29**	0/32

At the training-center placement, RCC-coordinate displacement minimization reduces Cα displacement by 5.77 Å relative to one-shot RCC, with a descriptive PDB-bootstrap 95% interval of -8.43 to -3.36 Å. Refolding RMSD also decreases by 6.13 Å (-9.31, -3.25). Compared with the infeasible raw LoopGen output, the optimized closure improves refolding RMSD by 1.45 Å (-2.61, -0.27).

The midpoint placement gives a different result. RCC-coordinate displacement minimization again reduces displacement from the generated trace, but its refolding RMSD is 1.71 Å higher than after one-shot RCC. Minimizing movement of the generated vertices is therefore not, by itself, a reliable biological objective. The experiment does show that closure can introduce substantial distortion and that, for the training-center placement, this distortion can be reduced while maintaining the geometric constraints exactly. None of the generated backbones passes the 2 Å refolding threshold, so this experiment does not establish sequence-structure self-consistency or biological design quality.

E.8 Endpoint-conditioned generation in RCC coordinates

To test whether the earlier refolding failures arise from RCC geometry or from the external generator and how its output is joined to the fixed framework, we evaluate a frozen Context-FrameChart stochastic interpolant that generates directly in the RCC bridge chart. The model conditions on H3 length, fixed framework anchors, endpoint and boundary tangents, masked sequence, and the visible framework/antigen context. It produces a connected Cα trace without a separate closure step. A learned residue-frame decoder reconstructs N/Cα/C/O atoms, and a synchronization step adjusts only the non-Cα atoms.

We select 32 distinct PDBs by identifier hash from the 112-condition development split before examining these results. This split is PDB-disjoint from the 875 conditions used to train the generator. For each condition, we retain two unranked backbone samples and draw two ProteinMPNN H3 sequences per backbone. Paired controls use the same generated Cα trace with either the unsynchronized atom decoder or four-flank PULCHRA reconstruction. We also evaluate native-backbone redesign and

native-sequence controls with the same ABodyBuilder3 pipeline. All 544 evaluations complete successfully.

Table 28: Refolding of endpoint-conditioned H3 samples on 32 development PDBs. Generated rows contain two unranked backbones per PDB and two sequence designs per backbone. Native-backbone redesign uses two sequence samples per PDB, and the native-sequence control uses one. Contact retention additionally requires a nonempty generated contact set and contact F1 of at least 0.5. These model-based criteria do not establish experimental folding or binding.

Backbone input	Attempts	RMSD ≤ 2 Å	PDBs ≥ 1	Contact-retaining	PDBs ≥ 1
Context-FrameChart, synchronized atoms	128	24 (18.75%)	11/32	16 (12.50%)	9/32
Context-FrameChart, raw atoms	128	28 (21.88%)	12/32	20 (15.62%)	10/32
Same generated Cα, PULCHRA atoms	128	28 (21.88%)	11/32	20 (15.62%)	9/32
Native atoms, H3 redesign	64	21 (32.81%)	11/32	15 (23.44%)	8/32
Native Cα, PULCHRA atoms, H3 redesign	64	20 (31.25%)	11/32	14 (21.88%)	8/32
Native atoms and sequence	32	21 (65.62%)	21/32	18 (56.25%)	18/32

With synchronized atoms, 24 of 128 sequence attempts across 11 PDBs satisfy the 2 Å self-consistency criterion, and 16 attempts across nine PDBs also retain the required antigen contacts. Because some generated backbones pass these tests, exact RCC bridge geometry does not by itself explain the all-zero result in the earlier refolding pilot. The native-sequence control performs substantially better, while native-backbone redesign also fails on many examples. These controls show that the sequence-design and refolding pipeline is imperfect even when it starts from an observed backbone, and they do not distinguish errors in sequence design from errors in structure prediction.

The synchronization step does not improve downstream refolding on this panel. Its self-consistency and contact-retaining rates are each 3.125 percentage points below those of the raw atom decoder, with a PDB-clustered descriptive 95% interval of -8.59 to $+1.56$ points, and they are also below the PULCHRA control (-12.50 to $+4.69$ points). Mean H3 Cα self-consistency RMSD is 4.04 Å with synchronization, 3.95 Å with the raw decoder, and 4.06 Å after PULCHRA reconstruction. Synchronization does reduce the mean maximum graft C-N discrepancy from 0.244 to 0.126 Å, but this local improvement does not produce a measurable refolding benefit.

We also rerun the complete pipeline from the frozen inputs. All 64 generated samples and all 544 reported metrics are reproduced, and the raw and synchronized Cα coordinates are bitwise identical. Every generated trace satisfies the 10^{-5} Å link-length tolerance, with a maximum link error of 1.46×10^{-6} Å. These development results show that generation directly in RCC coordinates can produce backbones that pass the model-based self-consistency criterion, but they are not an independent confirmation. We have also not determined the degree of training-set homology overlap among the generator, sequence designer, and refolding model. None of these results measures physical folding, affinity, or binding probability.

Several cases illustrate the remaining sources of error. For 8HHY, even the native-sequence ABodyBuilder3 prediction places the heavy and light chains differently from the deposited structure. Supplying the complete deposited variable-domain sequences does not resolve the discrepancy: native-sequence H3 self-consistency RMSD remains approximately 25 Å. The large error for 8HHY cannot be attributed to the generated H3 loop without first checking the reference chain geometry and a second structure predictor.

Sequence choice also changes the measured outcome. On six of the 64 primary generated backbones, one ProteinMPNN sequence passes the self-consistency criterion while the other fails, despite both sequences being designed for the same backbone. Redesigned sequences also fail on some native-backbone controls. More sequence samples and an independent predictor would be needed to separate sequence compatibility from prediction error.

Generated-backbone quality remains another source of failure. None of the 56 primary sequence attempts for loops of length at least 15 passes the self-consistency threshold. In addition, 7MZG, 3VG9, and 7X63 pass both native-backbone redesign controls but fail all four generated-backbone attempts. These cases point to limitations of the generated backbone, or its interaction with sequence design and refolding, that cannot be explained by failure of the native controls alone. The association with loop length is descriptive. Separating generator quality, sequence choice, and prediction error

will require matched comparisons across backbones, sequences, and structure predictors; isolating the contribution of RCC itself will require a paired comparison in which the constraint representation is varied while the rest of the pipeline is held fixed.

F ALLOCATION CONFIRMATION

We train a predictor that maps a scalar load to an allocation over 32 units, using the exact solution of a separable quadratic dispatch problem as the training target. The training loss does not include a penalty on the total allocation. We fix the predictor, RCC compiler, Euclidean projection baseline, evaluation metrics, and decision thresholds before generating 8,192 new evaluation loads.

Table 29: Bounded-allocation correction on 8,192 unseen loads. The frozen predictor allocates a fixed total across 32 units. RMSE and cost regret are measured relative to the exact separable quadratic optimum. Projection work reports the median and range of bisection iterations, while RCC uses a single fixed pass. Bold indicates the lower value between the two feasible correction methods.

Interface	Max total error ↓	Capacity violations	Median RMSE ↓	Median cost regret ↓	Max idempotence RMS	Work
Raw predictor	0.1915	0	0.003588	-	-	-
Euclidean projection	9.98×10^{-13}	0	**0.003525**	**0.001095**	7.32×10^{-14}	44 (31, 46)
Recursive compiler	7.11×10^{-15}	0	0.003674	0.001222	1.01×10^{-15}	1 fixed

Euclidean projection gives the lower RMSE and cost regret on this task. Compared with the raw predictor, RCC increases median RMSE by 2.4%; compared with projection, it increases median cost regret by 11.6%. Both differences remain within the prespecified 25% margins. The goal here is transfer to a second constraint family, not superiority to a projection specialized to this allocation problem.

Table 30: Coordinate and density checks on 312 allocation fibers. The panel covers every $n = 2, \ldots, 40$ and eight budget ratios from 0.035 to 0.965. Entries show the maximum absolute discrepancy in float64. Volume and density checks use surface measure on the fixed-total hyperplane.

Check	Maximum error
Recursive versus SVD log volume	2.84×10^{-14}
SVD versus reduced-QR log volume	3.20×10^{-14}
SVD versus direct-Gram log volume	1.42×10^{-14}
Reversed-tree chart area identity	4.71×10^{-14}
Split-coordinate gradient	1.42×10^{-13}
Budget gradient	5.33×10^{-15}
Allocation total	3.55×10^{-15}
Encode-decode reconstruction	2.28×10^{-15}
Encoded-density identity	6.55×10^{-12}
Affine volume and density covariance	1.03×10^{-12}

All 23 prespecified checks pass, and every logistic-normal proposal sample is strictly feasible and invertible. This verifies the RCC coordinate and density calculations for the bounded fixed-total allocation family studied here. The experiment does not address learned versus uniform allocation laws, other polytope families, or whether RCC should replace Euclidean projection for this task.