Anchored Protein Engineering

Chi Zhang[1,*] **Maria Rosaria Briglia**[2,*] **Litu Rout**[1,*] **Jeffrey Ouyang-Zhang**[3]
Iacopo Masi[2] **Sanjay Shakkottai**[1] **Adam Klivans**[1] **Daniel Diaz**[1]

[1]The University of Texas at Austin [2]Sapienza University of Rome [3]Massachusetts Institute of Technology
[*]Equal contribution.

Abstract

Inspired by recent work on discrete diffusion with anchored remasking, we introduce a training-free framework for protein engineering that generates higher-order variants that improve a target phenotype while accounting for epistatic interactions among the mutations. Prior approaches typically choose mutation sites without reward information and can drive sequences away from the natural protein manifold. We address these limitations with **A**nchored **P**rotein **E**ngineering via quantized e**X**pectation (**APEX**), an inference-time method consisting of anchor selection, tilted decoding, and optional iterative refinement. APEX selects anchor residues using a reward-gradient score with an explicit correction for pairwise interactions, decodes substitutions at the anchors by tilting the base model's per-site predictive distributions along reward gradients, and revisits low-scoring anchors iteratively for optional test-time scaling. All phases query the base model only through its per-position residue probabilities, a primitive shared by masked language models, discrete-diffusion models, and inverse-folding networks; as a result, APEX works out of the box across commonly used model families. Across thermostability, solubility, and GB1 binding benchmarks, APEX outperforms the compared baselines in predicted gain while avoiding the loss of sequence and structural plausibility that reward-search-only baselines tend to incur.

1 Introduction

A major goal of ML-guided protein engineering is to design higher-order variants: sequences carrying several coordinated mutations that improve a target property while preserving the functional context of the source protein. Such variants can deliver larger gains in stability, solubility, binding, or activity than single substitutions, but they are harder to design because mutational effects depend on context. Edits that help in isolation can be neutral or harmful in combination, and some favorable effects appear only when several sites change together. Directed evolution, deep mutational scanning, and display-based technologies have transformed protein engineering [Arnold, 2018, Romero and Arnold, 2009, Fowler and Fields, 2014], yet even high-throughput assays sample a small fraction of the combinatorial sequence space, and much of a limited experimental budget goes to variants that fail to fold, function, or improve the target. The computational problem is to propose mutation sets that make larger, property-improving sequence jumps while keeping the risk of nonfunctional variants low.

Pretrained generative protein models are a strong prior for this search. Masked language models [Lin et al., 2023, Hayes et al., 2025], discrete-diffusion models [Wang et al., 2024, Alamdari et al., 2023], and inverse-folding networks [Dauparas et al., 2022, Hsu et al., 2022] encode the evolutionary and structural context that makes a substitution plausible. Plausibility alone is not engineering, though. A method also needs the target signal to decide *where* to edit and *what* to put there; it must account for interactions among the chosen sites; and it must respect a fixed mutation budget, since wet-lab cost

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

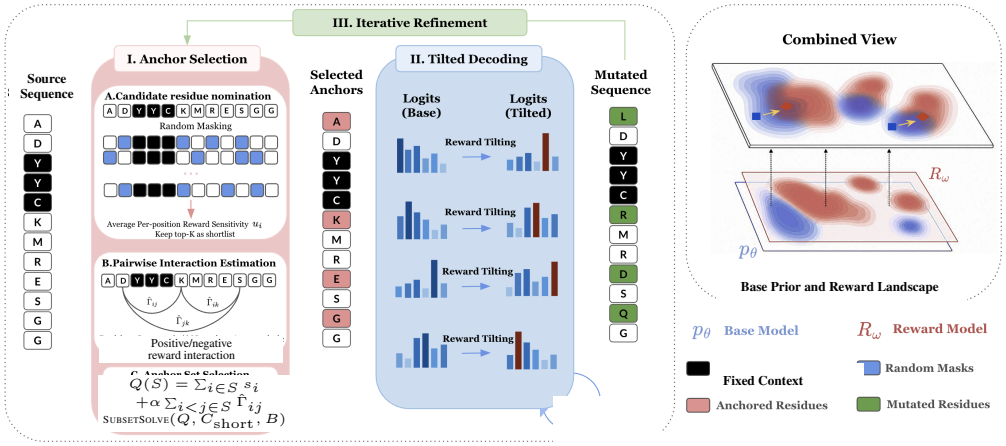

Figure 1: **The APEX pipeline.** Given a source sequence, a frozen base model, and a reward model, Phase I screens candidate residues with reward gradients and selects anchors using single-site scores and pairwise reward interactions (*where* to mutate). Phase II tilts the base model's amino-acid probabilities toward higher predicted reward and commits a substitution at each anchor (*what* to substitute). Optional refinement revisits weak edits without changing the mutation budget. Right: anchoring reward optimization to the base prior keeps designs on the protein manifold while improving the target property.

grows with the number of edits. Optimizing a learned predictor can also reward the predictor's own artifacts, so designs need complementary sequence- and structure-based checks.

Existing methods cover parts of this problem. Generative models [Madani et al., 2023, Wang et al., 2024, Dauparas et al., 2022] propose context-aware sequences but usually pick edit locations with reward-agnostic heuristics, which rank positions by model uncertainty rather than by their effect on the target and say nothing about which sites work well together. Reward-guided search such as PEX and GGS [Ren et al., 2022, Kirjner et al., 2024] optimizes a surrogate directly in sequence space, and PPDE [Emami et al., 2023] adds task-specific experts to a protein-language-model prior; in our experiments these methods raise the predicted property but lower sequence likelihood and structural confidence on the way. Activation steering (SPLM) [Huang et al., 2025] pushes internal representations along a property direction, and in our runs its substitutions collapse onto a few residues (Figure 3). What is missing is a method that selects sites with the reward, models how they interact, controls the edit count exactly, and still decodes from the base model's own residue distributions.

We introduce **A**nchored **P**rotein **E**ngineering via quantized e**X**pectation (**APEX**), an inference-time method for editing a source protein under a fixed mutation budget. Given the source sequence, a differentiable reward model for the target property, and a budget B, APEX chooses which residues to mutate and which amino acids to place there (Figure 1). Phase I screens candidate positions with reward gradients and selects *anchor residues* from single-site scores and an explicit pairwise reward-interaction term. Phase II adapts the quantized-expectation guidance of anchored posterior sampling [Rout et al., 2026]: it starts from the base model's amino-acid probabilities, tilts them toward higher predicted reward, and commits discrete substitutions. An optional refinement stage revisits weak edits without changing the budget. Because every phase reads the frozen base model only through its per-position residue probabilities, the same procedure runs on masked language models, discrete-diffusion models, and inverse-folding networks without retraining. On the primary thermostability and solubility benchmarks (the low-starting-property splits) APEX has the largest predicted gains and success rates of the compared methods, and it is the only reward-guided method whose designs stay as likely as their sources under the protein language model, with the smallest losses in structural confidence and evolutionary compatibility. A retrospective GB1 lookup and a T4 lysozyme case study show that the sites and substitutions it selects overlap with experimentally characterized beneficial mutations.

2 Related Work

Anchored posterior sampling. Anchored posterior sampling (APS) [Rout et al., 2026] is a test-time guidance method for discrete diffusion that builds on masked- and score-based discrete diffusion [Sahoo et al., 2024, Lou et al., 2024, Shi et al., 2024, Nie et al., 2025, Ye et al., 2025]. APEX inherits two of its ingredients. One is straight-through quantized-expectation guidance, in which the forward pass projects an expected embedding onto the amino-acid codebook and the backward pass differentiates through the soft expectation. The other is anchored remasking, which commits a high-confidence site and remasks the rest. APEX adds what a fixed-budget editor of an existing protein requires on top of this guidance: reward-aware site selection with pairwise interaction scores, exact control of the mutation count, and decoding schedules for each base-model family.

Inference-time guidance and sampling. Discrete Guidance derives exact guidance for discrete diffusion and flow models through continuous-time rate matrices [Nisonoff et al., 2025], and ProteinGuide conditions masked, autoregressive, diffusion, and flow-based protein models on auxiliary properties without retraining [Xiong et al., 2025]. Both guide the generation of a whole sequence. APEX instead edits a few positions of a given sequence under an exact budget, which means it also has to decide which positions to edit. In concurrent work, McCarter et al. [2026] search one-edit neighborhoods of masked language models with stochastic beam search and pseudo-perplexity; APEX follows reward gradients and selects sites jointly.

Reward-guided protein optimization. Several methods guide sequence search with learned fitness signals. PEX [Ren et al., 2022] couples a learned surrogate with Hamming-proximity proposals, and GGS [Kirjner et al., 2024] fits a graph-diffused CNN surrogate and samples mutations from its smoothed landscape. PPDE/EvoProtGrad [Emami et al., 2023] combines a pretrained sequence prior with task-specific experts, and SPLM [Huang et al., 2025] steers protein-language-model activations along property directions. APEX trains neither a proposal model nor a steering module: it selects the sites under an exact budget, starts each decoding step from the frozen base model's amino-acid probabilities, and lets reward gradients choose the substitution, so every edit stays tied to the base model's local preferences. App. D discusses further related work.

3 Method

Let $x^{(0)} \in \mathcal{V}^L$ be a source sequence of length L over the 20-amino-acid alphabet $\mathcal{V}$, with $[L] = \{1, \dots, L\}$ and editable positions $U \subseteq [L]$. Given a frozen protein model p_θ, a frozen differentiable reward model R_ω, and an exact mutation budget B, we seek

$$\max_{\substack{S \subseteq U,\ |S|=B \\ a_i \in \mathcal{V} \setminus \{x_i^{(0)}\},\ i \in S}} R_\omega\left(x^{(0)}[S \leftarrow a_S]\right), \tag{1}$$

where $x^{(0)}[S \leftarrow a_S]$ places residues a_S at positions S. The non-source constraint guarantees exactly B substitutions. Exhaustive search spans $\binom{|U|}{B} 19^B$ designs, so APEX separates *where* to edit from *what* to substitute. Phase I selects a size-B anchor set, Phase II decodes a substitution at each anchor, and an optional refinement stage reconsiders the weakest edits at constant budget (Figure 1). Throughout, APEX reads p_θ only through its per-position residue probabilities, which is why one procedure serves several model families. Derivations and model-specific details are collected in App. C.

3.1 Reward-Coupled Formulation

A reward-coupled objective makes the two roles explicit. Let $\mathcal{S}_B(U) = \{S \subseteq U : |S| = B\}$, let μ_B be a prior over anchor sets, and let p_θ^S be the base model's proposal over non-source assignments on S under a fixed decoding schedule. Writing $R_B(S, a_S) = R_\omega(x^{(0)}[S \leftarrow a_S])$ and $\pi_{\theta,B}(S, a_S) = \mu_B(S)\, p_\theta^S(a_S \mid x^{(0)}, \mathcal{G})$ for the joint prior, with optional structural context $\mathcal{G}$, a distribution q over feasible edits trades reward against deviation from this prior:

$$\mathcal{J}_B(q) = \mathbb{E}_q[R_B(S, a_S)] - \beta\, \mathrm{KL}(q \| \pi_{\theta,B}), \qquad \beta > 0. \tag{2}$$

Factoring $q(S, a_S) = \rho(S)\, q_S(a_S)$ into a choice of anchor set and a choice of substitutions given the set, the KL chain rule gives

$$\mathcal{J}_B(q) = \sum_{S \in \mathcal{S}_B(U)} \rho(S) \Big[\mathbb{E}_{q_S} R_B(S, a_S) - \beta\, \mathrm{KL}(q_S \| p_\theta^S) \Big] - \beta\, \mathrm{KL}(\rho \| \mu_B). \tag{3}$$

For each anchor set the bracket is maximized by the reward-tilted proposal $q_\beta(a_S \mid S) \propto p_\theta^S(a_S) \exp\big(R_B(S, a_S)/\beta\big)$, with optimal value $V_\beta(S)$; with a uniform μ_B, committing to a single anchor set reduces the outer problem to $\max_S V_\beta(S)$ up to a constant. APEX follows this structure: rank anchor sets, then tilt the base proposal toward the reward on the chosen set. We treat the objective as a reference rather than something we optimize exactly. Phase I ranks anchor sets with the gradient-based scores below, and Phase II starts from the base logits and takes reward-only steps, so the trade-off β belongs to the formulation and is unrelated to the softmax temperature τ used in decoding.

Base-model residue probabilities. For queried positions M and a site $i \in M$, the frozen base model returns logits ℓ_i^M and probabilities

$$p_i^M(a) = \mathrm{softmax}\big(\ell_i^M(x, \mathcal{G})\big)\,(a), \tag{4}$$

where x is the current sequence. ESM-2 and DPLM return all queried positions from one masked forward pass; LigandMPNN and SolubleMPNN obtain the same quantity by placing each queried site first in an autoregressive decoding order, which costs $|M|$ evaluations. Each model's probabilities are reordered to the reward model's alphabet before use, and no gradient passes through p_θ.

Straight-through quantized expectation. APEX adapts the quantized-expectation guidance of APS [Rout et al., 2026], a straight-through estimator (STE). Let $\mathrm{Emb}(a)$ be the reward model's embedding of amino acid a, and let $h_i = \mathrm{Emb}(\mathrm{NN}(\bar{e}_i))$ be the codebook embedding nearest to $\bar{e}_i$ in Euclidean distance. For optimizable logits ψ_i and softmax temperature $\tau > 0$, define

$$q_i = \mathrm{softmax}(\psi_i/\tau), \qquad \bar{e}_i = \textstyle\sum_a q_i(a)\, \mathrm{Emb}(a), \\ e_i^{\mathrm{STE}} = \bar{e}_i + \mathrm{stopgrad}(h_i - \bar{e}_i)\,. \tag{5}$$

The reward model therefore receives a discrete amino-acid embedding in the forward pass, while gradients flow through the soft expectation $\bar{e}_i$ and guide the residue choice (Figure 2) [Bengio et al., 2013, Van Den Oord et al., 2017, Rout et al., 2026]; stopgrad is the identity in the forward pass and has zero derivative in the backward pass.

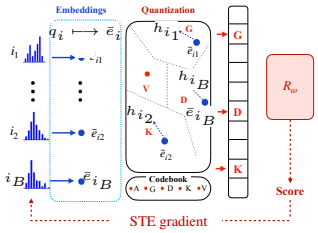

Figure 2: Straight-through quantized expectation. The reward model receives the nearest codebook embedding in the forward pass; gradients flow back through the soft expected embedding.

Supported base models. The same two-phase procedure applies to masked language models, discrete-diffusion models, and structure-conditioned inverse-folding models. Each family supplies the probabilities in Eq. 4 in its own way and decodes in its own order; the guidance gradients always come from the reward model, so the optimization logic is shared. The two phases are summarized in the boxes below.

Phase I: Anchor Selection

Input: Source $x^{(0)}$, budget B, editable set U, models p_θ, R_ω, screening settings N_{mc}, T, shortlist size K_{short}, pair weight α.
Output: Anchors $S^\star \subseteq U$, $|S^\star| = B$.

Estimate each screening score u_i over sampled contexts.
Retain the K_{short} largest scores in C_{short}.
Compute $\hat{s}_i$ and $\hat{\Gamma}_{ij}$ on C_{short}.
Set $s_i \leftarrow \hat{s}_i$ by default; form $Q(S)$.
$S^\star \leftarrow \textsc{SubsetSolve}(Q, C_{\mathrm{short}}, B)$.
return $S^\star$.

Phase II: Anchored Tilted Decoding

Input: Source $x^{(0)}$, anchors $S^\star$, models p_θ, R_ω, steps T_{opt}, step size η, temperature τ, decoding order m.
Output: Mutant $x^\star$ with B substitutions.

$x^\star \leftarrow x^{(0)}$; $F \leftarrow S^\star$.
while $F \neq \varnothing$ **do**
  Query base logits for the next site(s) in order m.
  Take T_{opt} straight-through reward steps.
  Exclude the source residue at each active site.
  Decode under m; update $x^\star$, F, and context.
end while
return $x^\star$.

3.2 Anchor Selection

Residues differ in how much the reward moves when they are substituted, and a substitution at one residue can change the best choice at another. Phase I accounts for both when choosing where to edit: a cheap gradient-based screen ranks all editable positions, single-site and pairwise reward probes then score a shortlist, and a subset solver picks the size-B anchor set.

Phase I-A: context-averaged screening. We draw N_{mc} screening contexts $C \subseteq U$ of T positions each. Within a context, every position is allowed to move toward the base model's prediction at the same time, so a position is scored while its neighbors are in flux too. For $i \in C$, the base-model probabilities p_i^C define the expected embedding and its straight-through counterpart,

$$\bar{e}_i^C = \sum_{a \in \mathcal{V}} p_i^C(a)\mathrm{Emb}(a), \qquad \widetilde{e}_i^C = \bar{e}_i^C + \mathrm{stopgrad}\left(\mathrm{Emb}\left(\arg\max_a p_i^C(a)\right) - \bar{e}_i^C\right). \tag{6}$$

With $g_i^C = \nabla_{e_i} R_\omega$ the reward gradient when every $j \in C$ carries $\widetilde{e}_j^C$, the screening score is the first-order reward change of moving residue i from its source identity toward the base model's expectation:

$$u_i(C) = \sum_{a \in \mathcal{V}} p_i^C(a) \left\langle g_i^C, \mathrm{Emb}(a) - \mathrm{Emb}(x_i^{(0)}) \right\rangle = \left\langle g_i^C, \bar{e}_i^C - \mathrm{Emb}(x_i^{(0)}) \right\rangle. \tag{7}$$

Averaging over the contexts that contain i gives u_i, and the $K_{\mathrm{short}} = 2B$ positions with the largest u_i form the shortlist C_{short}.

Phase I-B: single-site and pair scores. Each shortlisted position is queried on its own (a singleton mask for masked-token models; first in the decoding order for inverse-folding models), giving a proposal p_i and expected embedding $\bar{e}_i = \sum_a p_i(a)\mathrm{Emb}(a)$. Let

$$R_0 = R_\omega(x^{(0)}), \qquad r_i = R_\omega\left(x^{(0)}[i \leftarrow \bar{e}_i]\right), \qquad r_{ij} = R_\omega\left(x^{(0)}[\{i,j\} \leftarrow (\bar{e}_i, \bar{e}_j)]\right),$$

where replacement by an embedding acts on the reward model's input embeddings with all other positions fixed to the source. The single-site score and the pairwise reward interaction are

$$\hat{s}_i = r_i - R_0, \qquad \hat{\Gamma}_{ij} = r_{ij} - r_i - r_j + R_0. \tag{8}$$

$\hat{\Gamma}_{ij}$ is the departure of the joint effect from the sum of the single effects: positive values mark pairs that cooperate under the reward model, negative values pairs that interfere. It is a proxy computed from soft probes of the learned reward, not a measurement of biological epistasis.

Phase I-C: subset selection. With s_i the single-site term ($s_i = \hat{s}_i$ for the thermostability and solubility benchmarks, $s_i = u_i$ for the T4 case study) and α the pair weight, a candidate set scores

$$Q(S) = \sum_{i \in S} s_i + \alpha \sum_{\substack{i<j \\ i,j \in S}} \hat{\Gamma}_{ij}. \tag{9}$$

The solver enumerates all $\binom{K_{\mathrm{short}}}{B}$ subsets when this is small (12,870 at $B = 8$) and otherwise uses greedy selection followed by single-site swaps; the pair weight is ablated in §4.5. Because R_0 and the singleton probes are reused across pairs, Phase I-B costs $N_{\mathrm{probe}} = 1 + K_{\mathrm{short}} + \binom{K_{\mathrm{short}}}{2}$ batched reward evaluations, and screening confines this quadratic cost to the shortlist.

What the pair term captures. Hold the singleton embeddings fixed and let $\mathcal{F}(S) = R_\omega(x^{(0)}[S \leftarrow \bar{e}_S])$ with $\mathcal{F}(\varnothing) = R_0$. Then $\hat{s}_i$ and $\hat{\Gamma}_{ij}$ are the first and second finite differences of $\mathcal{F}$, and for $s_i = \hat{s}_i$ and $\alpha = 1$,

$$Q(\{i\}) = \mathcal{F}(\{i\}) - \mathcal{F}(\varnothing), \qquad Q(\{i,j\}) = \mathcal{F}(\{i,j\}) - \mathcal{F}(\varnothing). \tag{10}$$

Q thus reproduces the soft reward change exactly for one- and two-site sets; for larger sets it keeps the first- and second-order terms, and interactions among three or more sites remain as a residual. The identity concerns fixed soft proposals and says nothing by itself about the reward of the final discrete design.

3.3 Anchored Tilted Decoding

Phase II chooses the amino acid at each anchor. Let F be the anchors not yet decoded and $x^\star$ the current partially edited sequence. The base-model logits initialize $\psi_i^{(0)} = \ell_i^F(x^\star)$, so decoding starts on the protein prior rather than on a uniform distribution. For masked-token models the straight-through reward over the open sites is

$$\widetilde{R}_F(\psi_F) = R_\omega\big(x^\star[F \leftarrow e^{\mathrm{STE}}(\psi_F)]\big)\,, \tag{11}$$

and we take T_{opt} reward-ascent steps $\psi_F^{(t+1)} = \psi_F^{(t)} + \eta\,\widehat{\nabla}_{\psi_F}\widetilde{R}_F(\psi_F^{(t)})$ with step size η, where $\widehat{\nabla}$ is the straight-through pseudo-gradient. Let $g_i = \nabla_{e_i} R_\omega$ be evaluated at the hard forward input. Differentiating the soft expectation gives the coordinate-wise update direction

$$\Big[\widehat{\nabla}_{\psi_i}\widetilde{R}_F\Big]_a = \frac{q_i(a)}{\tau}\,\langle g_i, \mathrm{Emb}(a) - \bar{e}_i\rangle\,, \tag{12}$$

which raises the logit of every residue whose embedding lies further along the local reward gradient than the current expectation, in proportion to its current probability. The residue distributions factorize across anchors, but the full-sequence reward couples their updates. Because of the hard projection in the forward pass this is a pseudo-gradient, not an exact derivative through the discrete choice.

To guarantee exactly one substitution per anchor, the source residue is excluded before decoding:

$$\widetilde{\psi}_i(a) = \begin{cases} -\infty, & a = x_i^{(0)}, \\ \psi_i^{(T_{\mathrm{opt}})}(a), & \text{otherwise.} \end{cases} \tag{13}$$

The decoding schedule follows the base-model family. With ESM-2, the thermostability and solubility experiments update all anchors jointly and decode them in parallel by argmax. DPLM commits the most confident substitution, remasks the rest, and queries the model again, so later substitutions condition on earlier ones. Inverse-folding models decode in a fixed autoregressive order.

3.4 Test-Time Scaling: Optional Iterative Refinement

A single pass can leave weak edits in place: the anchor ranking and the substitution decoding are both approximate, and early substitutions are chosen against a partial context. Refinement keeps the strongest edits, ranked by their cached marginal contribution to Q, reverts the other positions to the source residues, and reruns anchor selection and decoding with the kept edits fixed in context. A schedule shrinks the number of reconsidered positions from round to round, so the search grows at inference time while the mutation budget stays fixed.

4 Experiments

We compare APEX with reward-guided baselines [Ren et al., 2022, Emami et al., 2023, Kirjner et al., 2024, Huang et al., 2025] and with random mutations on thermostability, solubility, and GB1 binding, then turn to a T4 lysozyme case study and ablations. The setting is one-shot: each method receives a source, a budget B, and a frozen reward and emits a design without an acquisition loop, so the comparison is per-design quality at a matched budget rather than query efficiency. Each thermostability and solubility split uses 100 distinct source proteins with one design per source and method, and results are grouped by the requested budget. APEX uses 16 random screening partitions with $T = B$, a shortlist of $K_{\mathrm{short}} = 2B$, pair weight $\alpha = 1$, softmax temperature $\tau = 1$, and 32 gradient steps with $\eta = 1.0$ in a single selection–decoding pass; App. A gives the full protocol and the baseline settings.

Reward models and plausibility metrics. The thermostability reward is an ESM-2 LoRA regressor trained on Meltome melting temperatures following the TemBERTure recipe [Rodella et al., 2024]; it reaches Spearman 0.886 on the held-out TemBERTure test split, slightly above the published three-replica ensemble (0.879). The solubility reward is trained on DeepSol [Khurana et al., 2018] following Huang et al. [2025], and GB1 uses ProteinNPT [Notin et al., 2023a]. Every final design in a task is scored with the same evaluator, including designs proposed with a different internal surrogate, and reported changes are design minus source. A method can raise a learned reward by drifting

away from natural sequences, so each table pairs the reward with proxies that do not depend on it. $\Delta\text{BLOSUM}/\text{mut}$ and the StructureDCA energy change $\Delta E/\text{mut}$ measure evolutionary compatibility per mutation; the residue-averaged ESM-2-650M pseudo-log-likelihood (PLL), ESMFold pLDDT (global and at edited sites), and backbone TM-score to the source (BB-TM) measure sequence and structure quality. They catch different failure modes; none measures the target property itself.

4.1 Thermostability

Thermostability is a central engineering objective because stable proteins are easier to express, purify, assay, and optimize further. We evaluate the first 100 proteins of the low-stability split of Huang et al. [2025] (mean length 212 ± 32 residues) at $B \in \{8, 16\}$. Methods that need a generative prior use ESM-2-650M, and success means a predicted $\Delta T_m \geq 5\,^\circ\text{C}$. The high-stability split is in Table 5.

Table 1: Predicted thermostability on THERM-LOW, $B \in \{8, 16\}$, $N=100$ per method. APEX uses ESM-2-650M. Reward is the predicted melting temperature; the remaining columns are reward-independent plausibility proxies. THERM-HIGH results are in Table 5.

	Reward		Evolutionary Compatibility		Sequence/Structure Quality		
Method	T_m (°C, Δ)	Succ% ↑	$\frac{\Delta\text{BLOSUM}}{\text{mut}}$ ↑	$\frac{\Delta E}{\text{mut}}$ ↓	PLL ↑	pLDDT (global/mut) ↑	BB-TM ↑
THERM-LOW, $B=8$							
Source	46.13	—	—	—	-1.82	80.8 / —	—
Random Mutations	48.37 (+2.24)	20	-6.53	+6.70	-1.97 (-0.15)	79.1 (-1.7) / 75.9	0.888
PEX [Ren et al., 2022]	54.51 (+8.38)	64	-7.19	+6.68	-1.99 (-0.17)	77.9 (-2.8) / 71.4	0.866
PPDE [Emami et al., 2023]	52.19 (+6.06)	56	-6.66	+6.58	-1.96 (-0.14)	79.3 (-1.5) / 74.0	0.888
GGS [Kirjner et al., 2024]	53.06 (+6.93)	61	-6.48	+6.66	-1.98 (-0.15)	78.5 (-2.3) / 71.6	0.881
SPLM [Huang et al., 2025]	53.52 (+7.39)	58	-6.95	+6.52	-1.94 (-0.12)	79.5 (-1.3) / 72.5	0.874
APEX (ESM-2-650M base)	**55.40 (+9.27)**	**70**	**-5.93**	**+4.06**	**-1.75 (+0.07)**	**80.8 (+0.0) / 79.4**	**0.899**
THERM-LOW, $B=16$							
Source	46.13	—	—	—	-1.82	80.8 / —	—
Random Mutations	48.24 (+2.10)	23	-6.60	+6.72	-2.14 (-0.32)	76.4 (-4.4) / 73.4	0.850
PEX [Ren et al., 2022]	57.92 (+11.79)	84	-7.07	+6.70	-2.16 (-0.35)	75.8 (-5.0) / 70.5	0.825
PPDE [Emami et al., 2023]	55.25 (+9.11)	73	-6.71	+6.58	-2.11 (-0.29)	76.6 (-4.2) / 72.6	0.835
GGS [Kirjner et al., 2024]	55.90 (+9.76)	73	-6.52	+6.42	-2.10 (-0.28)	76.3 (-4.5) / 70.6	0.848
SPLM [Huang et al., 2025]	56.63 (+10.50)	78	-7.11	+6.40	-2.05 (-0.23)	77.6 (-3.2) / 71.3	0.827
APEX (ESM-2-650M base)	**60.57 (+14.43)**	**87**	**-5.75**	**+3.93**	**-1.70 (+0.12)**	**80.3 (-0.5) / 78.8**	**0.864**

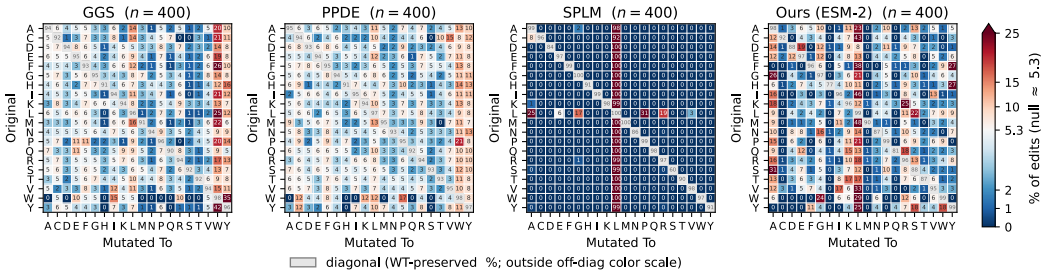

Figure 3: Amino-acid substitution patterns pooled across both thermostability splits and budgets ($n=400$ designs per method). Off-diagonal entry (i, j) is the percentage of edits from source residue i to target residue j, normalized within each edited source residue; gray diagonal entries show the percentage of source residues left unchanged. SPLM collapses onto leucine and GGS and PPDE over-use tryptophan and tyrosine, whereas APEX's substitutions are chemically conservative and track the residue being replaced.

APEX gives the best trade-off between predicted stability gain and plausibility (Table 1). At $B=8$ it raises the mean predicted melting temperature by $9.27\,^\circ\text{C}$ with 70% success; at $B=16$ the gain is $14.43\,^\circ\text{C}$ with 87% success, $2.6\,^\circ\text{C}$ and three points ahead of the strongest baseline at that budget. The gains are not bought by pushing sequences into regions that only the reward model likes. APEX is the only reward-guided method whose designs are more likely under ESM-2 than their sources (ΔPLL $+0.07$ and $+0.12$). Its global pLDDT stays within half a point of the source where the baselines lose 1–5 points, its edited residues keep 5–8 more pLDDT points than those of the reward-guided baselines, and its StructureDCA energy change per mutation is about 40% lower.

Figure 3 shows where the difference comes from. The reward-driven baselines raise the predicted property through narrow compositional biases: SPLM puts leucine at almost every edited site, and GGS and PPDE lean on tryptophan and tyrosine. APEX's substitutions are chemically conservative and depend on the residue they replace (Table 10). Decoding from the base model's own residue distributions is what keeps its designs on the protein prior.

4.2 Solubility

Soluble expression is often a prerequisite for production and characterization. We evaluate the first 100 proteins of the low-solubility split of Huang et al. [2025] (mean length 160 ± 51 residues), report the predicted solubility probability P_{sol}, and count success as $\Delta P_{\text{sol}} \geq 0.10$. The high-solubility split is in Table 6.

Table 2: Predicted solubility on SOL-LOW, $B \in \{8, 16\}$, $N=100$ per method. Columns follow Table 1; SOL-HIGH results are in Table 6.

	Reward		Evolutionary Compatibility		Sequence/Structure Quality		
Method	P_{sol} (Δ)	Succ% ↑	$\frac{\Delta\text{BLOSUM}}{\text{mut}}$ ↑	$\frac{\Delta E}{\text{mut}}$ ↓	PLL ↑	pLDDT (global/mut) ↑	BB-TM ↑
SOL-LOW, $B=8$							
Source	0.079	—	—	—	-1.73	76.5 / —	—
Random Mutations	0.156 (+0.077)	27	-6.64	+6.39	-1.97 (-0.24)	73.6 (-2.9) / 70.4	0.875
PEX [Ren et al., 2022]	0.253 (+0.174)	57	-6.64	+5.87	-1.82 (-0.10)	75.3 (-1.2) / 69.0	0.872
PPDE [Emami et al., 2023]	0.263 (+0.184)	58	-6.91	+6.24	-2.00 (-0.27)	72.6 (-3.9) / 67.8	0.862
GGS [Kirjner et al., 2024]	0.246 (+0.167)	54	-6.72	+6.14	-1.96 (-0.23)	73.7 (-2.8) / 70.0	0.882
SPLM [Huang et al., 2025]	0.276 (+0.197)	61	-7.73	+6.47	-2.08 (-0.35)	71.9 (-4.6) / 68.8	0.838
APEX (ESM-2-650M base)	**0.300 (+0.221)**	**64**	**-6.11**	**+3.32**	**-1.66 (+0.07)**	**76.5 (+0.0) / 74.6**	**0.900**
SOL-LOW, $B=16$							
Source	0.079	—	—	—	-1.73	76.5 / —	—
Random Mutations	0.150 (+0.070)	32	-6.64	+6.09	-2.17 (-0.45)	70.5 (-6.0) / 66.6	0.828
PEX [Ren et al., 2022]	0.380 (+0.301)	76	-6.94	+5.96	-2.19 (-0.46)	69.8 (-6.7) / 64.7	0.810
PPDE [Emami et al., 2023]	0.309 (+0.230)	67	-6.73	+6.04	-2.16 (-0.44)	69.9 (-6.6) / 65.4	0.833
GGS [Kirjner et al., 2024]	0.331 (+0.252)	70	-6.61	+5.77	-2.14 (-0.41)	71.3 (-5.2) / 66.7	0.834
SPLM [Huang et al., 2025]	0.363 (+0.283)	72	-6.97	+6.59	-2.26 (-0.53)	65.9 (-10.6) / 62.1	0.780
APEX (ESM-2-650M base)	**0.405 (+0.326)**	**81**	**-6.08**	**+3.59**	**-1.75 (-0.02)**	**74.6 (-1.9) / 72.8**	**0.836**

The solubility results follow the same pattern (Table 2). APEX has the largest gains and success rates at both budgets, $+0.221$ with 64% success at $B=8$ and $+0.326$ with 81% at $B=16$, which is 0.025 and five points ahead of the strongest baseline at $B=16$. Sequence likelihood is essentially unchanged (ΔPLL $+0.07$ and -0.02) where every reward-guided baseline loses 0.10–0.53, and APEX again has the lowest StructureDCA energy change and the highest edited-site pLDDT and BB-TM at both budgets. SPLM at $B=16$ is the clearest failure case: a competitive gain of $+0.283$ comes with the largest PLL drop (-0.53) and a 10.6-point loss in global pLDDT, which is what one expects when a single activation direction is pushed hard against the constraints of the protein prior.

4.3 GB1 Binding

Binding affinity underpins therapeutic engineering. We evaluate the 55-residue B1 immunoglobulin-binding domain of *Streptococcal* protein G (GB1), a standard system for binding and epistasis [Gronenborn et al., 1991, Sauer-Eriksson et al., 1995, Olson et al., 2014, Wu et al., 2016], with ProteinNPT [Notin et al., 2023a] trained on the Olson deep mutational scanning (DMS) data as the reward. Edits are restricted to the assayed window (positions 228–282) and each method generates $N=100$ designs at $B=2$. Because the assay measures nearly every double mutant in this window, each design can be looked up for its measured log-enrichment $\ln W$; success is $\ln W > 1.0$, above the 90th percentile of all measured doublets.

APEX has the highest predicted fitness, and the measurements agree. It has the highest mean DMS fitness, and 98% of its designs clear $\ln W > 1.0$ (Table 3). It is also the only method whose edits lower the StructureDCA energy ($\Delta E/\text{edit} = -0.44$), with the highest PLL, edited-site pLDDT, and BB-TM. Since ProteinNPT was trained on this same DMS, the benchmark is a retrospective, in-distribution check that the chosen double mutants are measurably good, not an independent test of generalization; the fixed-budget search claims rest on the $B \in \{8, 16\}$ benchmarks above.

Table 3: GB1 binding on the Olson 2014 DMS, B=2, N=100 designs per method. Columns follow Table 1, except that the per-mutation proxies are reported per edit and success is defined on the measured DMS log-enrichment of each design. BB-TM is computed over the assayed 55-residue B1 domain (positions 228–282).

	Reward	Evolutionary Compatibility		Sequence/Structure Quality			Measured DMS	
Method	Fitness (Δ)	$\frac{\Delta\text{BLOSUM}}{\text{edit}}\uparrow$	$\frac{\Delta E}{\text{edit}}\downarrow$	PLL ↑	pLDDT (global/mut) ↑	BB-TM ↑	DMS lnW ↑	Succ% ↑
Source	-0.39	—	—	-1.38	69.0 / —	—	0.00	—
Random Mutations	-1.28 (-0.90)	-4.41	+3.10	-1.40 (-0.02)	68.6 (-0.4) / 77.2	0.948	-1.08	5
PEX [Ren et al., 2022]	0.93 (+1.31)	-6.00	+3.71	-1.40 (-0.02)	68.5 (-0.5) / 74.9	0.974	+0.80	67
PPDE [Emami et al., 2023]	0.67 (+1.06)	-6.22	+3.33	-1.41 (-0.03)	68.5 (-0.5) / 76.6	0.971	+0.96	52
GGS [Kirjner et al., 2024]	0.95 (+1.33)	-6.53	+3.67	-1.41 (-0.03)	68.1 (-0.9) / 74.0	0.964	+1.24	94
SPLM [Huang et al., 2025]	0.81 (+1.20)	-7.55	+4.17	-1.42 (-0.04)	68.6 (-0.4) / 80.7	0.983	+1.03	93
APEX (ESM-2-650M base)	**1.08 (+1.47)**	-6.46	**-0.44**	**-1.35 (+0.03)**	**68.7 (-0.3) / 86.2**	**0.997**	**+1.30**	**98**

4.4 Case Study: T4 Lysozyme

T4 lysozyme is a classic stability-engineering system [Matthews, 1995, Eriksson et al., 1992, Matsumura et al., 1989], which lets us ask whether the sites and substitutions APEX selects coincide with those found experimentally. Starting from the 164-residue cysteine-free C54T/C97A variant (PDB 2LZM), we compare APEX with PEX, GGS, PPDE+ESM-2, and Random Mutations at $B \in \{8, 16\}$ under the thermostability reward. We also report a T4-specific *Catalog-hit %*, the fraction of designs that contain at least one of 31 literature-associated stabilizing substitutions at 22 positions.

APEX has the largest mean predicted ΔT_m at both budgets ($+3.0\,^\circ$C at B=8, $+5.9\,^\circ$C at B=16) with BB-TM 0.997, and it recovers catalog substitutions most often, in 32% and 60% of its designs at B=8 and B=16 against 14–28% and 30–48% for the reward-guided baselines (Table 7). Its edits thus overlap with stabilizing mechanisms that have been characterized experimentally, not only with what the reward model scores highly. A catalog hit does not by itself establish the stability or function of a full multi-mutant design. Because the evaluated design sets differ between methods, we treat this as a descriptive case study rather than a controlled comparison (App. A.4).

4.5 Ablations and Analysis

Table 4: **APEX ablation experiments.** All results use THERM-LOW, $B = 8$, $N = 100$. Gray cells mark default settings; Table 8 reports the full results.

(a) **Base model.** The same procedure transfers across model families.

Backbone	$\Delta R\uparrow$	$\Delta E/\text{mut}\downarrow$	ΔPLL ↑	BB-TM ↑
LigandMPNN	5.73	5.64	-0.08	**0.93**
SolubleMPNN	5.74	5.60	-0.07	**0.93**
DPLM-650M	8.88	4.20	+0.06	0.90
ESM-2-650M	**9.27**	**4.06**	**+0.07**	0.90

(b) **Phases.** Both phases outperform either alone.

I	II	$\Delta R\uparrow$	$\Delta E/\text{mut}\downarrow$	ΔPLL ↑
	✓	3.03	4.95	-0.01
✓		6.98	**4.02**	**+0.07**
✓	✓	**9.27**	4.06	**+0.07**

(c) **Pair interactions.** The pairwise term improves anchor selection.

Pair weight α	$\Delta R\uparrow$	$\Delta E/\text{mut}\downarrow$	ΔPLL ↑
0.0	7.34	**3.98**	+0.06
1.0	**9.27**	4.06	**+0.07**

(d) **Iterative refinement.** A second pass adds a modest gain.

Config	$\Delta R\uparrow$	$\Delta E/\text{mut}\downarrow$	ΔPLL ↑
Single round	9.27	**4.06**	**+0.07**
With refinement	**9.77**	4.11	**+0.07**

Table 4 ablates APEX on THERM-LOW at B=8; Table 8 repeats every ablation on all four splits, and the text quotes both.

APEX is base-model agnostic. Table 4a runs the same procedure on a masked language model (ESM-2-650M), a discrete-diffusion model (DPLM-650M), and two inverse-folding models. ESM-2 and DPLM give the largest reward gains with positive ΔPLL, and DPLM trails ESM-2 by only 0.4–0.5 °C and at most 0.007 in P_{sol} across the four splits. LigandMPNN and SolubleMPNN gain 22–71% as much, but they keep the backbone closest to the source (BB-TM 0.92–0.94 against 0.89–0.90

for ESM-2) and have the highest global pLDDT, so the choice of base model is a reward–structure trade-off. All four run through the same per-position-probability interface, each with its own decoding order.

Anchor selection drives most of the gain. Phase II alone, that is, reward-tilted decoding at random positions, reaches only $\Delta R = 3.03\,^{\circ}\mathrm{C}$ (Table 4b). Across the four splits it recovers at most 44% of the full gain and goes negative on THERM-HIGH ($-1.29\,^{\circ}\mathrm{C}$), and among the ESM-2 configurations it is the only one with negative ΔPLL and has the highest $\Delta E/\mathrm{mut}$ on every split. The reward can be climbed at unselected sites only by leaving the prior. Phase I alone, which selects anchors and then takes the base model's own substitutions, reaches $6.98\,^{\circ}\mathrm{C}$ and recovers 55–78% of the full gain on every split with the same ΔPLL as the full method. Together the two phases reach $9.27\,^{\circ}\mathrm{C}$, with ΔPLL and $\Delta E/\mathrm{mut}$ essentially unchanged ($+0.07$; 4.02 to 4.06). Site selection does most of the work; reward tilting adds a substantial gain on top of it at no measurable cost in plausibility.

The pair term captures useful interactions. Dropping the pairwise term ($\alpha = 0$) lowers ΔR from 9.27 to $7.34\,^{\circ}\mathrm{C}$ (Table 4c), and across the splits it costs 16–33% of the gain, most on the -HIGH splits (33% and 32%, against 21% and 16%). $\hat{\Gamma}_{ij}$ finds pairs whose joint substitution is favorable under the reward and that first-order selection misses, at little cost in plausibility; on SOL-LOW it even raises ΔPLL from $+0.02$ to $+0.07$.

Test-time scaling helps modestly. One refinement pass raises ΔR to $9.77\,^{\circ}\mathrm{C}$, with ΔPLL unchanged at $+0.07$ and $\Delta E/\mathrm{mut}$ moving only from 4.06 to 4.11 (Table 4d); on every split it adds 4–12% to the gain, leaves ΔPLL unchanged or higher, and moves BB-TM by at most 0.012. We treat it as an optional way to buy reward with compute and use a single pass by default.

The gains survive a change of predictor. The thermostability reward is one learned predictor, so we re-scored every $B{=}16$ design with the published TemBERTure regressor, a different architecture that no method optimized against, trained on the same Meltome-derived data (Table 9). APEX leads on both splits ($+6.59$ and $+7.92\,^{\circ}\mathrm{C}$), and its margin over the next-best method is wider under the held-out predictor than under the optimized one ($+3.7$ versus $+2.6\,^{\circ}\mathrm{C}$ on THERM-LOW; $+3.2$ versus $+0.6\,^{\circ}\mathrm{C}$ on THERM-HIGH). At the level of individual substitutions (Table 10), APEX is the only method whose target residue depends on the residue it replaces, with $+0.19$ to $+0.23$ bits of excess mutual information against at most $+0.06$ for any baseline. It also makes chemically conservative substitutions (BLOSUM62 ≥ 0) 40–48% of the time, against 25–30% for the best baseline. The stability gains hold under a predictor the method never saw, and on all four splits the substitutions are chemically coherent rather than the product of a compositional bias.

5 Conclusion

APEX designs higher-order protein variants under a fixed mutation budget at inference time. It selects anchor residues with reward gradients and an explicit pairwise-interaction correction, decodes substitutions by tilting the frozen base model's residue distributions along reward gradients, and optionally refines weak edits; no model is trained or modified in the process. Because it needs only per-position residue probabilities, the same procedure runs unchanged on all three model families we test. On the thermostability and solubility benchmarks it has the largest predicted gains of the compared methods while keeping sequence likelihood intact and losing far less structural confidence and evolutionary compatibility than the baselines. On GB1 and T4 lysozyme, the sites and substitutions it selects overlap with experimentally characterized beneficial mutations. These gains are predictions from learned evaluators; prospective experiments are still needed to confirm that the prioritized variants improve the underlying properties. Limitations and broader impacts are discussed in App. E. Code will be released at `https://github.com/skyzczzz/APEX`.

Acknowledgments and Disclosure of Funding

This work was supported by the NSF AI Institute for Foundations of Machine Learning (IFML) and the UT-Austin Center for Generative AI. This work has been partially supported by PRIN 2022 project 20227YET9B "AdVVent" CUP code B53D23012830006, and project BEAT (Better dEep leArning securiTy), a Sapienza University project.

Competing interests. Daniel J. Diaz owns Intelligent Proteins LLC, where he consults biotechnology companies on AI protein engineering. D.J.D. is a co-founder of Metabologic AI, which focuses on developing commercial enzymes with AI. The other authors declare no competing interests.

References

Frances H. Arnold. Directed evolution: Bringing new chemistry to life. *Angewandte Chemie International Edition*, 57(16):4143–4148, 2018. doi: 10.1002/anie.201708408.

Philip A. Romero and Frances H. Arnold. Exploring protein fitness landscapes by directed evolution. *Nature Reviews Molecular Cell Biology*, 10(12):866–876, 2009. doi: 10.1038/nrm2805. URL https://doi.org/10.1038/nrm2805.

Douglas M. Fowler and Stanley Fields. Deep mutational scanning: a new style of protein science. *Nature Methods*, 11(8):801–807, 2014. doi: 10.1038/nmeth.3027. URL https://doi.org/10.1038/nmeth.3027.

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

Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J. Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q. Tran, Jonathan Deaton, Marius Wiggert, Rohil Badkundri, Irhum Shafkat, Jun Gong, Alexander Derry, Raul S. Molina, Neil Thomas, Yousuf A. Khan, Chetan Mishra, Carolyn Kim, Liam J. Bartie, Matthew Nemeth, Patrick D. Hsu, Tom Sercu, Salvatore Candido, and Alexander Rives. Simulating 500 million years of evolution with a language model. *Science*, 387(6736):850–858, 2025. doi: 10.1126/science.ads0018. URL https://doi.org/10.1126/science.ads0018.

Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 52309–52333. PMLR, 2024. URL https://proceedings.mlr.press/v235/wang24ct.html.

Sarah Alamdari, Nitya Thakkar, Rianne van den Berg, Neil Tenenholtz, Robert Strome, Alan M. Moses, Alex X. Lu, Nicolo Fusi, Ava P. Amini, and Kevin K. Yang. Protein generation with evolutionary diffusion: sequence is all you need. *bioRxiv*, 2023. doi: 10.1101/2023.09.11.556673. URL https://doi.org/10.1101/2023.09.11.556673.

J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, A. Courbet, R. J. de Haas, N. Bethel, P. J. Y. Leung, T. F. Huddy, S. Pellock, D. Tischer, F. Chan, B. Koepnick, H. Nguyen, A. Kang, B. Sankaran, A. K. Bera, N. P. King, and D. Baker. Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, 378(6615):49–56, 2022. doi: 10.1126/science.add2187. URL https://doi.org/10.1126/science.add2187.

Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8946–8970. PMLR, 2022. URL https://proceedings.mlr.press/v162/hsu22a.html.

Ali Madani, Ben Krause, Eric R. Greene, Subu Subramanian, Benjamin P. Mohr, James M. Holton, Jose Luis Olmos, Jr., Caiming Xiong, Zachary Z. Sun, Richard Socher, James S. Fraser, and

Nikhil Naik. Large language models generate functional protein sequences across diverse families. *Nature Biotechnology*, 41(8):1099–1106, 2023. doi: 10.1038/s41587-022-01618-2. URL https://doi.org/10.1038/s41587-022-01618-2.

Zhizhou Ren, Jiahan Li, Fan Ding, Yuan Zhou, Jianzhu Ma, and Jian Peng. Proximal exploration for model-guided protein sequence design. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 18520–18536. PMLR, 2022. URL https://proceedings.mlr.press/v162/ren22a.html.

Andrew Kirjner, Jason Yim, Raman Samusevich, Shahar Bracha, Tommi Jaakkola, Regina Barzilay, and Ila Fiete. Improving protein optimization with smoothed fitness landscapes. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rxlF2Zv8x0.

Patrick Emami, Aidan Perreault, Jeffrey Law, David Biagioni, and Peter St. John. Plug & play directed evolution of proteins with gradient-based discrete MCMC. *Machine Learning: Science and Technology*, 4(2):025014, 2023. doi: 10.1088/2632-2153/accacd. URL https://doi.org/10.1088/2632-2153/accacd.

Long-Kai Huang, Rongyi Zhu, Bing He, and Jianhua Yao. Steering protein language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 26247–26260. PMLR, 2025. URL https://proceedings.mlr.press/v267/huang25ba.html.

Litu Rout, Andreas Lugmayr, Yasamin Jafarian, Srivatsan Varadharajan, Constantine Caramanis, Sanjay Shakkottai, and Ira Kemelmacher-Shlizerman. Test-time anchoring for discrete diffusion posterior sampling. In *Forty-third International Conference on Machine Learning*, 2026. URL https://arxiv.org/pdf/2510.02291.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32819–32848. PMLR, 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025. URL https://arxiv.org/pdf/2502.09992.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025. URL https://arxiv.org/abs/2508.15487.

Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=XsgHl54yO7.

Junhao Xiong, Ishan Gaur, Maria Lukarska, Hunter Nisonoff, Luke M. Oltrogge, David F. Savage, and Jennifer Listgarten. Proteinguide: On-the-fly property guidance for protein sequence generative models, 2025. URL https://arxiv.org/abs/2505.04823.

Calvin McCarter, Nick Bhattacharya, Sebastian W. Ober, and Hunter Elliott. How to make the most of your masked language model for protein engineering, 2026. URL https://arxiv.org/abs/2603.10302.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. URL https://arxiv.org/abs/1308.3432.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. URL https://arxiv.org/pdf/1711.00937.

Chiara Rodella, Symela Lazaridi, and Thomas Lemmin. TemBERTure: advancing protein thermostability prediction with deep learning and attention mechanisms. *Bioinformatics Advances*, 4 (1):vbae103, 2024. doi: 10.1093/bioadv/vbae103. URL https://doi.org/10.1093/bioadv/vbae103.

Sameer Khurana, Reda Rawi, Khalid Kunji, Gwo-Yu Chuang, Halima Bensmail, and Raghvendra Mall. DeepSol: a deep learning framework for sequence-based protein solubility prediction. *Bioinformatics*, 34(15):2605–2613, 2018. doi: 10.1093/bioinformatics/bty166. URL https://doi.org/10.1093/bioinformatics/bty166.

Pascal Notin, Ruben Weitzman, Debora S. Marks, and Yarin Gal. ProteinNPT: Improving protein property prediction and design with non-parametric transformers. In *Advances in Neural Information Processing Systems*, volume 36, 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6a4d5d85f7a52f062d23d98d544a5578-Abstract-Conference.html.

Angela M. Gronenborn, David R. Filpula, Norma Z. Essig, Anand Achari, Marc Whitlow, Paul T. Wingfield, and G. Marius Clore. A novel, highly stable fold of the immunoglobulin binding domain of streptococcal protein G. *Science*, 253(5020):657–661, 1991. doi: 10.1126/science.1871600. URL https://doi.org/10.1126/science.1871600.

A. Elisabeth Sauer-Eriksson, Gerard J. Kleywegt, Mathias Uhlén, and T. Alwyn Jones. Crystal structure of the C2 fragment of streptococcal protein G in complex with the Fc domain of human IgG. *Structure*, 3(3):265–278, 1995. doi: 10.1016/S0969-2126(01)00157-5. URL https://doi.org/10.1016/S0969-2126(01)00157-5.

C. Anders Olson, Nicholas C. Wu, and Ren Sun. A comprehensive biophysical description of pairwise epistasis throughout an entire protein domain. *Current Biology*, 24(22):2643–2651, 2014. doi: 10.1016/j.cub.2014.09.072. URL https://doi.org/10.1016/j.cub.2014.09.072.

Nicholas C. Wu, Lei Dai, C. Anders Olson, James O. Lloyd-Smith, and Ren Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife*, 5:e16965, 2016. doi: 10.7554/eLife.16965. URL https://doi.org/10.7554/eLife.16965.

Brian W. Matthews. Studies on protein stability with T4 lysozyme. *Advances in Protein Chemistry*, 46:249–278, 1995. doi: 10.1016/S0065-3233(08)60337-X. URL https://doi.org/10.1016/S0065-3233(08)60337-X.

A. E. Eriksson, W. A. Baase, X. J. Zhang, D. W. Heinz, M. Blaber, E. P. Baldwin, and B. W. Matthews. Response of a protein structure to cavity-creating mutations and its relation to the hydrophobic effect. *Science*, 255(5041):178–183, 1992. doi: 10.1126/science.1553543.

M. Matsumura, W. J. Becktel, M. Levitt, and B. W. Matthews. Stabilization of phage T4 lysozyme by engineered disulfide bonds. *Proceedings of the National Academy of Sciences*, 86(17):6562–6566, 1989. doi: 10.1073/pnas.86.17.6562.

Johannes Linder, Nicholas Bogard, Alexander B. Rosenberg, and Georg Seelig. A generative neural network for maximizing fitness and diversity of synthetic DNA and protein sequences. *Cell Systems*, 11(1):49–62.e4, 2020. doi: 10.1016/j.cels.2020.05.007.

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

Matsvei Tsishyn, Hugo Talibart, Marianne Rooman, and Fabrizio Pucci. Structure-informed direct coupling analysis improves protein mutational landscape predictions, 2026. URL https://www.biorxiv.org/content/10.64898/2026.03.27.714804v1. bioRxiv preprint.

Yang Zhang and Jeffrey Skolnick. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic Acids Research*, 33(7):2302–2309, 2005. doi: 10.1093/nar/gki524. URL https://doi.org/10.1093/nar/gki524.

Anna Jarząb, Nils Kurzawa, Thomas Hopf, et al. Meltome atlas—thermal proteome stability across the tree of life. *Nature Methods*, 17(5):495–503, 2020. doi: 10.1038/s41592-020-0801-4. URL https://doi.org/10.1038/s41592-020-0801-4.

Pascal Notin, Aaron Kollasch, Daniel Ritter, Lood van Niekerk, Steffanie Paul, Han Spinner, Nathan Rollins, Ada Shaw, Rose Orenbuch, Ruben Weitzman, Jonathan Frazer, Mafalda Dias, Dinko Franceschi, Yarin Gal, and Debora Marks. ProteinGym: Large-scale benchmarks for protein fitness prediction and design. In *Advances in Neural Information Processing Systems*, volume 36, 2023b. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/cac723e5ff29f65e3fcbb0739ae91bee-Abstract-Datasets_and_Benchmarks.html. Datasets and Benchmarks Track.

Will Grathwohl, Kevin Swersky, Milad Hashemi, David Duvenaud, and Chris Maddison. Oops I took a gradient: Scalable sampling for discrete distributions. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3831–3841. PMLR, 2021. URL https://proceedings.mlr.press/v139/grathwohl21a.html.

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

Casper A. Goverde, Martin Pacesa, Nicolas Goldbach, Lars J. Dornfeld, Petra E. M. Balbi, Sandrine Georgeon, Stéphane Rosset, Srajan Kapoor, Jagrity Choudhury, Justas Dauparas, Christian Schellhaas, Simon Kozlov, David Baker, Sergey Ovchinnikov, Alex J. Vecchio, and Bruno E. Correia. Computational design of soluble and functional membrane protein analogues. *Nature*, 631(8020):449–458, 2024. doi: 10.1038/s41586-024-07601-y. URL https://doi.org/10.1038/s41586-024-07601-y.

Noelia Ferruz, Steffen Schmidt, and Birte Höcker. ProtGPT2 is a deep unsupervised language model for protein design. *Nature Communications*, 13(1):4348, 2022. doi: 10.1038/s41467-022-32007-7. URL https://doi.org/10.1038/s41467-022-32007-7.

Christoffer Norn, Basile I. M. Wicky, David Juergens, Sirui Liu, David Kim, Doug Tischer, Brian Koepnick, Ivan Anishchenko, David Baker, and Sergey Ovchinnikov. Protein sequence design by conformational landscape optimization. *Proceedings of the National Academy of Sciences*, 118 (11):e2017228118, 2021. doi: 10.1073/pnas.2017228118. URL https://doi.org/10.1073/pnas.2017228118.

Sam Sinai, Richard Wang, Alexander Whatley, Stewart Slocum, Elina Locane, and Eric D. Kelsic. AdaLead: A simple and robust adaptive greedy search algorithm for sequence design, 2020. URL https://arxiv.org/abs/2010.02141.

Johannes Linder, Nicholas Bogard, Alexander B. Rosenberg, and Georg Seelig. Deep exploration networks for rapid engineering of functional DNA sequences, 2019. URL https://www.biorxiv.org/content/10.1101/864363v1. bioRxiv preprint; full version published as Linder et al. [2020].

Nate Gruver, Samuel Stanton, Nathan C. Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion. In *Advances in Neural Information Processing Systems*, volume 36, pages 12489–12517, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/29591f355702c3f4436991335784b503-Abstract-Conference.html.

Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. Model-based reinforcement learning for biological sequence design. In *International Conference on Learning Representations*, 2020.

Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure F. P. Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, Lena Simine, Payel Das, and Yoshua Bengio. Biological sequence design with GFlowNets. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9786–9801. PMLR, 2022.

David H. Brookes and Jennifer Listgarten. Design by adaptive sampling, 2018.

David Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 773–782. PMLR, 2019.

Anvita Gupta and James Zou. Feedback GAN for DNA optimizes protein functions. *Nature Machine Intelligence*, 1(2):105–111, 2019. doi: 10.1038/s42256-019-0017-4.

Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alexander Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/f51338d736f95dd42427296047067694-Abstract.html.

Jeffrey Ouyang-Zhang, Daniel J. Diaz, Adam R. Klivans, and Philipp Krähenbühl. Predicting a protein's stability under a million mutations. *Advances in Neural Information Processing Systems*, 36:76229–76247, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/f03cb785864596fa5901f1359d23fd81-Abstract-Conference.html.

Daniel J. Diaz, Chengyue Gong, Jeffrey Ouyang-Zhang, James M. Loy, Jordan Wells, David Yang, Andrew D. Ellington, Alexandros G. Dimakis, and Adam R. Klivans. Stability oracle: a structure-based graph-transformer framework for identifying stabilizing mutations. *Nature Communications*, 15(1):6170, 2024. doi: 10.1038/s41467-024-49780-2. URL https://doi.org/10.1038/s41467-024-49780-2.

Chengyue Gong, Adam R. Klivans, James Madigan Loy, Tianlong Chen, Qiang Liu, and Daniel Jesus Diaz. Evolution-inspired loss functions for protein representation learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*. PMLR, 2024. URL https://proceedings.mlr.press/v235/gong24e.html.

Bruce J. Wittmann, Yisong Yue, and Frances H. Arnold. Informed training set design enables efficient machine learning-assisted directed protein evolution. *Cell Systems*, 12(11):1026–1045.e7, 2021. doi: 10.1016/j.cels.2021.07.008. URL https://doi.org/10.1016/j.cels.2021.07.008.

Surojit Biswas, Grigory Khimulya, Ethan C. Alley, Kevin M. Esvelt, and George M. Church. Low-N protein engineering with data-efficient deep learning. *Nature Methods*, 18(4):389–396, 2021. doi: 10.1038/s41592-021-01100-y. URL https://doi.org/10.1038/s41592-021-01100-y.

Vincent Q. Tran, Matthew Nemeth, Liam J. Bartie, Sita S. Chandrasekaran, Alison Fanton, Hyungseok C. Moon, Brian L. Hie, Silvana Konermann, and Patrick D. Hsu. Rapid directed evolution guided by protein language models and epistatic interactions. *Science*, 0(0), 2026. doi: 10.1126/science.aea1820. URL https://www.science.org/doi/abs/10.1126/science.aea1820.

Gian Marco Visani, Aayush Verma, and William S. DeWitt. Additive baselines furnish no evidence for epistasis learning by multi-evolve. 2026. doi: 10.64898/2026.04.23.719915.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10835–10866. PMLR, 2023. URL https://proceedings.mlr.press/v202/gao23h.html.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/3d719fee332caa23d5038b8a90e81796-Abstract-Conference.html.

A Experimental setup

This section describes the evaluation metrics, reward models, baseline conventions, and task-specific settings used throughout the experiments.

A.1 Common protocol

APEX settings. The main thermostability and solubility experiments use ESM-2-650M as the frozen base model. Phase I uses 16 random partitions into screening contexts of size $T = B$, giving $N_{\mathrm{mc}} \approx 16L/B$ complete contexts when all positions are editable; the shortlist size is $K_{\mathrm{short}} = 2B$, and the pair weight is $\alpha = 1$. Phase II uses softmax temperature $\tau = 1$ and 32 gradient steps with step size $\eta = 1.0$. The main results use one Phase I–II pass without optional refinement, and all sequence positions are editable.

Reward and success rate. ΔR is the paired change $R(\text{design}) - R(\text{source})$ under the task-specific evaluator. It is measured in °C for predicted melting temperature, on a $[0, 1]$ scale for predicted solubility, and as log-enrichment (lnW) for GB1 binding. Success denotes $\Delta T_m \geq 5\,°\text{C}$ for thermostability, $\Delta P_{\mathrm{sol}} \geq 0.10$ for solubility, and measured $\mathrm{lnW} > 1.0$ for GB1. The GB1 threshold is above the 90th percentile ($\mathrm{lnW} \geq 0.88$) of the 535,917 measured double mutants [Olson et al., 2014].

Plausibility proxies. PLL is the per-residue pseudo-log-likelihood under an ESM-2-650M evaluator [Lin et al., 2023]; ΔPLL is the design value minus the source value. $\Delta\mathrm{BLOSUM}/\mathrm{mut}$ averages $\mathrm{BLOSUM62}(x_i^{(0)}, x_i^{\star}) - \mathrm{BLOSUM62}(x_i^{(0)}, x_i^{(0)})$ over edited positions [Henikoff and Henikoff, 1992]. $\Delta E/\mathrm{mut}$ is the mean per-edit energy change under source-specific, MSA-derived StructureDCA models [Tsishyn et al., 2026]; lower values are more favorable under this proxy. ESMFold supplies global and edited-site pLDDT, while BB-TM compares the predicted design and source backbones [Zhang and Skolnick, 2005]. These metrics probe complementary failure modes but do not directly measure the target property.

Compute. Optimization used a single NVIDIA H100 (80 GB) GPU. For THERM/SOL sources of 164–320 residues, one design at $B = 8$ took approximately 1–3 minutes. On GB1, the complete $B = 2$, $N = 100$ Phase I–II run for one base-model variant took approximately 30 minutes. Total project compute, including baselines and pilot configurations, was below 2,000 H100-GPU-hours.

GB1 artifact coverage. The archived window-restricted GB1 directory does not contain a completed output cohort. Consequently, the exact APEX generation settings underlying Table 3 cannot be reconstructed from the available artifacts; this is a reproducibility limitation of the retrospective analysis.

A.2 Reward models

The thermostability and solubility evaluators use an ESM-2-650M backbone with a property-specific supervised head. All designs in a task are scored by the same evaluator, whatever surrogate the method used internally.

- *Thermostability.* An ESM-2 LoRA model trained on 28,195 Meltome melting-temperature labels [Jarząb et al., 2020] following the TemBERTure recipe [Rodella et al., 2024]. On the 531-protein TemBERTure test split (measured T_m from 33 to 91 °C) it reaches Spearman 0.886, Pearson 0.894, RMSE 7.74 °C, and R^2 0.795, against 0.879, 0.887, 8.04 °C, and 0.778 for the published three-replica TemBERTure ensemble scored on the same proteins. Both predictors learn protein-level melting temperature across organisms rather than mutational ΔT_m, which is why every table pairs the reward with reward-independent proxies and why App. B.5 re-scores designs with the published regressor. The same checkpoint evaluates the T4 case study.
- *Solubility.* An ESM-2 LoRA model trained on 62,478 DeepSol labels [Khurana et al., 2018] following Huang et al. [2025].
- *GB1 binding.* A ProteinNPT model [Notin et al., 2023a] trained on the Olson 2014 DMS [Olson et al., 2014], accessed through ProteinGym [Notin et al., 2023b]. The model uses a random 10%

test split and obtains Spearman $\rho = 0.903$ on both validation and test sets. Because some designed double mutants may occur in its training split, the oracle column confirms their measured fitness but does not establish generalization to unseen double mutants.

All reward models remain frozen during design. APEX and SPLM differentiate through the relevant evaluator; PEX, GGS, and PPDE candidate pools are rescored with it.

A.3 Reward-driven baselines

Random Mutations samples B positions uniformly and replaces each source residue with a uniformly sampled alternative. SPLM [Huang et al., 2025] steers ESM-2 activations using property information. PEX [Ren et al., 2022] trains its own surrogate under the published procedure, after which we rescore its final designs with the shared evaluator. GGS [Kirjner et al., 2024] uses GWG sampling [Grathwohl et al., 2021], and PPDE [Emami et al., 2023] uses a product-of-experts energy containing an ESM-2 expert.

Methods are grouped by requested budget B. For SPLM, we sweep the steering coefficient $\gamma \in \{0.5, 1, 2, 3, 5, 10, 20\}$ over the published extraction recipe and balanced-quantile variants on each task's training corpus, then report the requested-budget setting with the largest shared-evaluator reward.

A.4 T4 case study

The source is the 164-residue cysteine-free C54T/C97A T4 lysozyme variant, with PDB 2LZM as the structural reference. The archived APEX artifacts use ESM-2-650M, unrestricted editable positions, the context-averaged score $s_i = u_i$, one joint reward tilt, and parallel argmax commitment; no MSA edit restriction or StructureDCA selection term is active. The $B = 8$ proxy-evaluation artifact records $T_{\text{opt}} = 32$, step size $\eta = 16$, and 16 seeds with 100 attempts each; deduplication leaves 53 distinct outputs. The $B = 16$ artifact records $T_{\text{opt}} = 32$, step size $\eta = 32$, and four seeds with 25 attempts each; its proxy evaluation contains 100 output rows representing 99 distinct sequences. Each baseline starts from 100 attempts under its native procedure.

The archived $B = 8$ reward, catalog-hit, and proxy summaries do not share one fully recoverable post-processing cohort. We therefore treat this single-source case study as descriptive rather than as a controlled head-to-head comparison.

The literature catalog includes a substitution when at least one reported variant containing it has a positive stability signal and at least one reported variant containing it retains function. Excluding the cysteine-free background substitutions, this gives 31 substitutions across 22 positions.

A.5 Licenses for existing assets

ESM-2 and ESMFold [Lin et al., 2023] use MIT terms; DPLM code [Wang et al., 2024] is Apache-2.0. LigandMPNN and SolubleMPNN [Dauparas et al., 2025, Goverde et al., 2024] use MIT terms for code and model parameters. ProteinNPT, ProteinGym, and StructureDCA software [Notin et al., 2023a,b, Tsishyn et al., 2026] are MIT-licensed. BLOSUM62 [Henikoff and Henikoff, 1992] and TM-score [Zhang and Skolnick, 2005] are credited at first use; the BB-TM implementation uses the GPLv3 `tmtools` wrapper and its MIT-licensed TM-align code. Meltome, DeepSol, and the Olson GB1 data remain subject to their upstream terms and are credited at first use. Exact historical revisions and some dataset and checkpoint terms remain unverified.

B Additional experimental results

This appendix reports the higher-starting-property splits, the solubility substitution heatmap, the full T4 results, the complete ablation breakdown, a held-out predictor check, and a substitution-level analysis.

B.1 Higher-starting-property splits

Tables 5 and 6 report the -HIGH splits at $B \in \{8, 16\}$, using the same metrics as the main-text tables. These sources begin with higher predicted target values and therefore leave less headroom for improvement.

Table 5: Predicted thermostability on THERM-HIGH, $B \in \{8, 16\}$, with $N = 100$ designs per method. Columns follow Table 1.

	Reward		Evolutionary Compatibility		Sequence/Structure Quality		
Method	T_m (°C, Δ)	Succ% ↑	$\frac{\Delta \text{BLOSUM}}{\text{mut}}$ ↑	$\frac{\Delta E}{\text{mut}}$ ↓	PLL ↑	pLDDT (global/mut) ↑	BB-TM ↑
THERM-HIGH, B=8							
Source	59.26	—	—	—	-1.66	80.4 / —	—
Random Mutations	55.88 (-3.38)	9	-6.55	+6.44	-1.89 (-0.23)	78.2 (-2.1) / 75.3	0.896
PEX [Ren et al., 2022]	59.34 (+0.08)	24	-6.80	+6.53	-1.74 (-0.09)	79.3 (-1.0) / 73.9	**0.934**
PPDE [Emami et al., 2023]	61.61 (+2.35)	33	-6.66	+6.32	-1.87 (-0.21)	78.0 (-2.3) / 73.1	0.901
GGS [Kirjner et al., 2024]	63.55 (+4.30)	45	-6.35	+6.26	-1.86 (-0.20)	78.2 (-2.1) / 73.0	0.898
SPLM [Huang et al., 2025]	61.29 (+2.03)	36	-7.39	+6.34	-1.88 (-0.22)	78.5 (-1.9) / 75.1	0.900
APEX (ESM-2-650M base)	**66.60 (+7.34)**	**63**	**-5.98**	**+2.99**	**-1.58 (+0.08)**	**80.1 (-0.2) / 78.0**	0.902
THERM-HIGH, B=16							
Source	59.26	—	—	—	-1.66	80.4 / —	—
Random Mutations	55.47 (-3.79)	10	-6.51	+6.17	-2.05 (-0.39)	75.2 (-5.1) / 71.6	0.857
PEX [Ren et al., 2022]	71.77 (+12.51)	74	-7.54	+6.36	-1.90 (-0.25)	77.6 (-2.7) / 69.2	0.868
PPDE [Emami et al., 2023]	66.16 (+6.90)	60	-6.79	+6.30	-2.06 (-0.40)	75.0 (-5.4) / 69.7	0.858
GGS [Kirjner et al., 2024]	67.69 (+8.43)	65	-6.51	+6.14	-2.04 (-0.38)	74.9 (-5.5) / 69.4	0.853
SPLM [Huang et al., 2025]	65.22 (+5.96)	50	-7.45	+6.13	-2.08 (-0.42)	75.1 (-5.2) / 72.1	0.840
APEX (ESM-2-650M base)	**72.36 (+13.10)**	**80**	**-5.91**	**+3.07**	**-1.60 (+0.06)**	**78.6 (-1.7) / 76.3**	**0.871**

Table 6: Predicted solubility on SOL-HIGH, $B \in \{8, 16\}$, with $N = 100$ designs per method. Columns follow Table 2.

	Reward		Evolutionary Compatibility		Sequence/Structure Quality		
Method	P_{sol} (Δ)	Succ% ↑	$\frac{\Delta \text{BLOSUM}}{\text{mut}}$ ↑	$\frac{\Delta E}{\text{mut}}$ ↓	PLL ↑	pLDDT (global/mut) ↑	BB-TM ↑
SOL-HIGH, B=8							
Source	0.277	—	—	—	-1.78	77.1 / —	—
Random Mutations	0.274 (-0.002)	29	-6.59	+6.33	-1.96 (-0.18)	74.4 (-2.7) / 70.7	0.877
PEX [Ren et al., 2022]	0.445 (+0.168)	37	-6.96	+6.19	-1.86 (-0.08)	76.3 (-0.8) / 70.6	0.847
PPDE [Emami et al., 2023]	0.392 (+0.116)	47	-6.66	+6.07	-1.93 (-0.15)	74.9 (-2.2) / 68.4	0.878
GGS [Kirjner et al., 2024]	0.384 (+0.108)	45	-6.56	+5.96	-1.92 (-0.14)	74.8 (-2.3) / 68.5	0.867
SPLM [Huang et al., 2025]	0.307 (+0.031)	33	-7.07	+6.52	-1.99 (-0.21)	72.7 (-4.4) / 71.1	0.866
APEX (ESM-2-650M base)	**0.457 (+0.181)**	**60**	**-6.17**	**+3.65**	**-1.71 (+0.07)**	**76.7 (-0.4) / 73.4**	**0.885**
SOL-HIGH, B=16							
Source	0.277	—	—	—	-1.78	77.1 / —	—
Random Mutations	0.252 (-0.025)	21	-6.61	+6.28	-2.15 (-0.37)	70.4 (-6.7) / 66.5	0.824
PEX [Ren et al., 2022]	0.547 (+0.271)	**84**	-7.30	+5.99	-1.96 (-0.18)	74.5 (-2.6) / 65.7	0.828
PPDE [Emami et al., 2023]	0.430 (+0.153)	54	-6.74	+6.19	-2.09 (-0.31)	71.6 (-5.5) / 66.9	0.835
GGS [Kirjner et al., 2024]	0.480 (+0.204)	62	-6.66	+6.01	-2.09 (-0.31)	71.8 (-5.3) / 66.4	0.835
SPLM [Huang et al., 2025]	0.548 (+0.272)	40	-6.89	+6.82	-2.21 (-0.43)	66.7 (-10.4) / 62.5	0.773
APEX (ESM-2-650M base)	**0.549 (+0.273)**	76	**-6.05**	**+3.63**	**-1.69 (+0.09)**	**75.6 (-1.5) / 73.4**	**0.842**

Reading the -HIGH splits. On SOL-HIGH at $B = 16$, APEX gains $+0.273$, compared with $+0.272$ for SPLM and $+0.271$ for PEX. PEX has the highest success rate (84%), while APEX retains the strongest reported plausibility proxies. The main text focuses on the lower-starting-property splits, where more improvement is possible.

B.2 Solubility substitution heatmap

Figure 4 gives the solubility counterpart of Figure 3, pooled over both splits and budgets ($n = 400$ designs per method).

B.3 T4 case study: full results

The T4 source and generation settings are summarized in §4.4 and App. A.4.

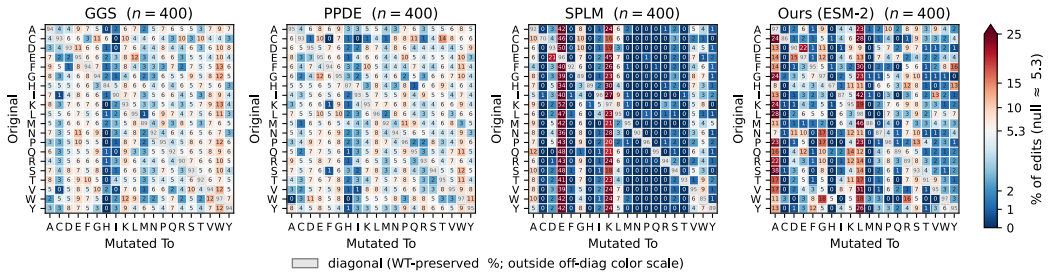

Figure 4: Amino-acid substitution patterns pooled across both solubility splits and budgets ($n = 400$ designs per method). Rows are source residues and columns are target residues; normalization follows Figure 3.

Table 7: T4 in silico results at $B = 8$ and $B = 16$. Archived evaluation counts vary by method and metric; the APEX $B = 8$ plausibility proxies use 53 distinct retained outputs, while the cohort behind its reward and catalog-hit summaries is not fully recoverable. Lower $\Delta E/\text{mut}$ is more favorable under StructureDCA. Catalog-hit is the fraction of designs containing a literature-associated stabilizing substitution.

	Reward	Evolutionary Compatibility		Sequence/Structure Quality			Literature
Method	T_m (°C, Δ)	$\frac{\Delta\text{BLOSUM}}{\text{mut}}$ ↑	$\frac{\Delta E}{\text{mut}}$ ↓	PLL ↑	pLDDT (global/mut) ↑	BB-TM ↑	Catalog-hit % ↑
(a) B=8							
Source	54.40	—	—	-0.32	89.7 / —	—	—
APEX	**57.42 (+3.02)**	**-4.49**	**0.14**	**-0.30 (+0.02)**	**89.6 / 86.8**	**0.997**	**32.0**
PEX	56.43 (+2.03)	-6.44	2.29	-0.52 (-0.20)	89.0 / 84.1	0.993	14.0
GGS	56.13 (+1.73)	-5.67	2.46	-0.47 (-0.16)	88.0 / 80.0	0.995	20.0
PPDE+ESM-2	55.65 (+1.25)	-6.25	2.28	-0.64 (-0.32)	88.5 / 83.1	0.994	28.0
Random Mutations	53.41 (-0.99)	-6.51	3.40	-0.82 (-0.50)	87.8 / 83.7	0.984	4.0
(b) B=16							
Source	54.40	—	—	-0.32	89.7 / —	—	—
APEX	**60.31 (+5.91)**	**-4.25**	**0.66**	**-0.43 (-0.11)**	**90.2 / 89.7**	**0.997**	**60.0**
PEX	59.14 (+4.74)	-6.39	3.03	-0.72 (-0.40)	88.3 / 83.0	0.987	48.0
GGS	58.09 (+3.69)	-5.66	2.83	-0.47 (-0.15)	87.6 / 78.3	0.995	30.0
PPDE+ESM-2	58.26 (+3.86)	-6.25	2.81	-0.87 (-0.55)	87.1 / 81.8	0.987	39.0
Random Mutations	52.44 (-1.96)	-6.33	3.80	-1.24 (-0.92)	84.7 / 80.7	0.975	16.0

Reward and plausibility. APEX has the largest mean predicted thermostability gains in the table: $+3.02\,°\text{C}$ at $B = 8$ and $+5.91\,°\text{C}$ at $B = 16$. It also has the most favorable BLOSUM62, StructureDCA, and predicted-structure proxy values in these cells. In the archived artifacts, StructureDCA is used only as a retrospective plausibility proxy and does not enter anchor selection. Catalog hits occur in 32% and 60% of its designs at the two budgets.

B.4 Full ablation breakdown

Table 8 reports every main-text ablation on all four splits, with the remaining proxies. All values use $B = 8$ and $N = 100$ designs per split.

B.5 Held-out predictor check

Because the thermostability reward is a single learned predictor, we also re-scored every B=16 design on both thermostability splits with the published TemBERTure regressor [Rodella et al., 2024], a ProtBERT-BFD model with adapters that no method optimized against. Both predictors are trained on Meltome-derived data, so this is a check across architectures rather than against independent labels. Table 9 reports the mean change in predicted melting temperature per design under the held-out predictor. APEX leads both splits, and its margin over the next-best method is larger under the held-out predictor than under the optimized reward of Tables 1 and 5: $+3.7$ versus $+2.6\,°\text{C}$ on THERM-LOW and $+3.2$ versus $+0.6\,°\text{C}$ on THERM-HIGH. Random Mutations is negative under the held-out predictor on both splits.

Table 8: Full ablation results. ΔR is measured in °C for THERM and as P_{sol} for SOL; therm-h/therm-l and sol-h/sol-l denote the -HIGH and -LOW splits. Lower $\Delta E/\text{mut}$ is more favorable under StructureDCA; ΔPLL is design minus source under ESM-2-650M; pLDDT is global. Bold marks the default configuration, not the best value.

Config	Split	ΔR	$\frac{\Delta E}{\text{mut}} \downarrow$	ΔPLL	pLDDT	BB-TM
Default APEX *(ESM-2-650M, α=1, single round)*						
APEX	therm-h	**+7.34**	**+2.99**	**+0.08**	**80.1**	**0.902**
	therm-l	**+9.27**	**+4.06**	**+0.07**	**80.8**	**0.899**
	sol-h	**+0.181**	**+3.65**	**+0.07**	**76.7**	**0.885**
	sol-l	**+0.221**	**+3.32**	**+0.07**	**76.5**	**0.900**
A1. *Phase decomposition: anchor selection supplies most of the reward gain.*						
Phase II only	therm-h	−1.29	+4.36	−0.03	79.5	0.922
	therm-l	+3.03	+4.95	−0.01	80.5	0.908
	sol-h	+0.027	+4.64	−0.03	76.5	0.903
	sol-l	+0.097	+4.42	−0.04	76.0	0.907
Phase I only	therm-h	+5.75	+2.93	+0.08	80.6	0.902
	therm-l	+6.98	+4.02	+0.07	80.7	0.899
	sol-h	+0.099	+3.66	+0.07	76.5	0.885
	sol-l	+0.168	+3.31	+0.07	76.4	0.900
A2. *Pairwise reward interaction improves selection.*						
α=0	therm-h	+4.93	+3.10	+0.07	80.0	0.918
	therm-l	+7.34	+3.98	+0.06	80.8	0.900
	sol-h	+0.123	+3.84	+0.06	76.6	0.892
	sol-l	+0.186	+3.63	+0.02	75.6	0.883
A3. *Iterative refinement yields modest gains.*						
With refinement	therm-h	+7.66	+2.95	+0.10	80.4	0.914
	therm-l	+9.77	+4.11	+0.07	80.7	0.903
	sol-h	+0.203	+3.70	+0.09	76.3	0.876
	sol-l	+0.239	+3.25	+0.09	76.4	0.895
A4. *Choice of base model.*						
DPLM-650M	therm-h	+6.88	+2.99	+0.08	80.2	0.919
	therm-l	+8.88	+4.20	+0.06	80.7	0.902
	sol-h	+0.174	+3.76	+0.06	76.3	0.878
	sol-l	+0.220	+3.50	+0.06	76.0	0.901
LigandMPNN	therm-h	+1.61	+5.13	−0.11	81.2	0.933
	therm-l	+5.73	+5.64	−0.08	81.5	0.927
	sol-h	+0.089	+5.00	−0.09	77.9	0.920
	sol-l	+0.156	+4.73	−0.10	77.5	0.924
SolubleMPNN	therm-h	+1.77	+5.01	−0.12	81.2	0.937
	therm-l	+5.74	+5.60	−0.07	81.4	0.932
	sol-h	+0.091	+4.99	−0.10	77.7	0.924
	sol-l	+0.157	+4.81	−0.11	77.5	0.921

B.6 Substitution-level analysis

Figures 3 and 4 suggest that the methods differ in how their substitutions depend on the residue being replaced. To quantify this, we pool the edits of each method on each split at B=16 (100 designs per cell), tabulate (source residue, target residue) pairs, and compute the mutual information between the two in excess of a permutation null that preserves both marginals and the no-self-substitution constraint (200 shuffles). The excess mutual information is zero both for a method that writes one residue everywhere and for one that draws targets uniformly; it is positive only when the replacement tracks the residue being replaced, as a substitution matrix does. Table 10 also reports the target-residue entropy, the fraction of chemically conservative substitutions (BLOSUM62 ≥ 0), and ΔPLL.

APEX is the only method whose substitutions depend substantially on what they replace, with $+0.19$ to $+0.23$ bits of excess mutual information against at most $+0.06$ for any baseline on any split. It makes 40–48% conservative substitutions against 25–30% for the best baseline on each split, and it

Table 9: Held-out predictor check at B=16, N=100 designs per method and split: mean change in melting temperature (°C) predicted by the published TemBERTure regressor, design minus source. No method optimized against this predictor.

Method	THERM-LOW	THERM-HIGH
Random Mutations	−0.84	−2.81
PEX [Ren et al., 2022]	+1.73	+4.74
PPDE [Emami et al., 2023]	+1.24	+2.12
GGS [Kirjner et al., 2024]	+1.93	+3.74
SPLM [Huang et al., 2025]	+2.88	+4.61
APEX (ESM-2-650M base)	**+6.59**	**+7.92**

has the highest ΔPLL on every split. Its target-residue entropy (3.5–3.8 bits) is below that of most baselines (3.1–4.3 bits) but far above SPLM's collapse (0.1–1.8 bits): APEX spends its diversity on substitutions that are individually plausible rather than on coverage of amino-acid space.

Table 10: Substitution-level statistics at B=16 (N=100 designs per cell). $H(\mathrm{tgt})$ is the entropy of the target-residue distribution in bits; excess I is the mutual information between source and target residue in excess of a marginal-preserving permutation null; B62≥0 is the percentage of substitutions with non-negative BLOSUM62 score.

Split	Method	$H(\mathrm{tgt})$	Excess I (bits)	B62≥0 (%)	ΔPLL
THERM-LOW	Random Mutations	4.31	+0.02	25.5	−0.32
	PEX	4.07	+0.03	17.6	−0.35
	PPDE	4.23	+0.01	22.0	−0.29
	GGS	4.16	+0.02	26.2	−0.28
	SPLM	0.12	+0.04	3.2	−0.23
	APEX	3.77	**+0.21**	**47.6**	**+0.12**
THERM-HIGH	Random Mutations	4.32	−0.02	24.9	−0.39
	PEX	3.05	+0.05	5.9	−0.25
	PPDE	4.17	+0.01	20.9	−0.40
	GGS	4.04	+0.03	22.4	−0.38
	SPLM	0.15	+0.04	4.5	−0.42
	APEX	3.54	**+0.23**	**44.2**	**+0.06**
SOL-LOW	Random Mutations	4.31	−0.02	24.7	−0.45
	PEX	4.18	+0.02	21.6	−0.46
	PPDE	4.27	+0.02	24.4	−0.44
	GGS	4.21	+0.03	25.3	−0.41
	SPLM	1.66	+0.03	29.8	−0.53
	APEX	3.72	**+0.19**	**40.5**	**−0.02**
SOL-HIGH	Random Mutations	4.31	−0.01	26.8	−0.37
	PEX	3.79	+0.06	16.8	−0.18
	PPDE	4.26	−0.01	24.0	−0.31
	GGS	4.20	+0.01	25.4	−0.31
	SPLM	1.78	+0.06	28.9	−0.43
	APEX	3.67	**+0.20**	**43.6**	**+0.09**

C Derivations and implementation details

Throughout, replacing a residue by an embedding-valued quantity acts on the corresponding input embedding of R_ω. In Phase I the soft embedding $\bar{e}_i$ is formed from base-model probabilities; in Phase II it is formed from the optimized probabilities q_i. Table 11 summarizes the notation.

Table 11: Notation used in the main text and appendix.

Symbol	Meaning
$x^{(0)}, x^\star$	Source sequence and final edited sequence.
$L, \mathcal{V}, U, B$	Sequence length, amino-acid alphabet, editable positions, and exact mutation budget.
$S, S^\star, F$	Candidate anchor set, selected anchor set, and anchors not yet decoded.
M, C, T, N_{mc}	Queried positions, a screening context, its size, and the number of screening contexts.
$C_{\mathrm{short}}, K_{\mathrm{short}}$	Shortlisted positions and shortlist size.
$p_\theta, \ell_i^M, p_i^M$	Frozen base model and its logits and amino-acid probabilities at position i.
$R_\omega, \mathrm{Emb}(a), \mathcal{G}$	Frozen reward model, its embedding of amino acid a, and optional structural context.
$\mu_B, \pi_{\theta,B}$	Anchor-set prior and joint prior over feasible edits.
ρ, q_S, q	Anchor-set, conditional assignment, and joint design distributions.
$\beta, V_\beta(S)$	KL trade-off parameter and optimal conditional objective of an anchor set.
$\mathcal{F}(S), d(W)$	Soft-replacement reward of a set and its inclusion–exclusion coefficient.
$\bar{e}_i, \widetilde{e}_i^C, h_i, e_i^{\mathrm{STE}}$	Soft expected embedding, its straight-through counterpart in screening, nearest hard embedding, and straight-through embedding.
g_i, g_i^C	Reward gradient with respect to the input embedding at position i, evaluated at the hard forward input (within screening context C).
$u_i, \hat{s}_i, \hat{\Gamma}_{ij}$	Screening score, single-site score, and pairwise reward interaction.
$s_i, \alpha, Q(S)$	Single-site term used for selection, pair weight, and anchor-set score.
$\psi_i, q_i, \tau, \eta, T_{\mathrm{opt}}, m$	Optimized logits, residue probabilities, softmax temperature, step size, number of reward steps, and decoding order.

C.1 Reward-coupled objective

Gibbs optimizer. For finite rewards and $\beta > 0$, the maximizer of Eq. 2 over distributions on feasible edits is the reward-tilted prior

$$q_B^\star(S, a_S) = \frac{1}{Z_B}\, \pi_{\theta,B}(S, a_S) \exp\!\left(\frac{R_B(S, a_S)}{\beta}\right), \qquad Z_B = \sum_{S', a'_{S'}} \pi_{\theta,B}(S', a'_{S'}) \exp\!\left(\frac{R_B(S', a'_{S'})}{\beta}\right), \tag{14}$$

where the sum runs over feasible anchor sets and their non-source assignments. Substituting $\log(q_B^\star/\pi_{\theta,B}) = R_B/\beta - \log Z_B$ into Eq. 2 gives $\mathcal{J}_B(q) = \beta \log Z_B - \beta\, \mathrm{KL}(q \| q_B^\star)$ for every q with finite KL, and the KL term vanishes only at $q = q_B^\star$. Applied to a single anchor set S, the same identity shows that the bracket in Eq. 3 is maximized by the tilted proposal $q_\beta(a_S \mid S) = p_\theta^S(a_S) \exp\big(R_B(S, a_S)/\beta\big)/Z_S$, with optimal value $V_\beta(S) = \beta \log Z_S$.

Decomposition. For $q(S, a_S) = \rho(S)\, q_S(a_S)$ with finite KL, $\log(q/\pi_{\theta,B}) = \log(\rho/\mu_B) + \log(q_S/p_\theta^S)$ on the support of q. Taking expectations under q gives the chain rule

$$\mathrm{KL}(q \| \pi_{\theta,B}) = \mathrm{KL}(\rho \| \mu_B) + \sum_S \rho(S)\, \mathrm{KL}(q_S \| p_\theta^S), \tag{15}$$

and substituting into Eq. 2 yields Eq. 3. Optimizing each q_S leaves $\sum_S \rho(S) V_\beta(S) - \beta\, \mathrm{KL}(\rho \| \mu_B)$. For uniform μ_B and a deterministic choice $\rho = \delta_S$, the outer KL equals $\log \binom{|U|}{B}$ for every S, so the best deterministic anchor set maximizes $V_\beta(S)$. APEX uses this structure as motivation only: Phase I ranks anchor sets with $Q(S)$, and Phase II starts from the base logits and takes reward-only steps instead of sampling q_β. Each p_θ^S is normalized over non-source assignments for a fixed decoding schedule; no consistency across schedules is required.

C.2 Base-model residue probabilities

For a current sequence x, queried positions M, and $i \in M$, Eq. 4 is shorthand for $p_i^M(a) = p_\theta(a_i = a \mid x_{-M}, \mathcal{G})$, where $\mathcal{G}$ supplies structural context when the model requires it. We reorder each

model's probabilities to the reward model's ordering of the 20 amino acids, and no gradient passes through p_θ. ESM-2 and DPLM replace the queried positions with mask tokens and return all of their logits from one forward pass. LigandMPNN and SolubleMPNN are autoregressive given a backbone, so we choose decoding orders that place each queried position first among the editable positions in turn; $|M|$ queried positions cost $|M|$ ordered evaluations. All families therefore provide the probabilities in Eq. 4, although they decode in different orders (App. C.4).

C.3 Phase I details

Screening. Each screening round partitions the editable positions at random into complete blocks of size T, dropping incomplete tails; the thermostability and solubility experiments use 16 rounds with $T = B$, which gives $N_{\mathrm{mc}} \approx 16L/B$ contexts. All positions in a block share one masked base-model query and one batched reward-gradient evaluation. For a position i with sampled contexts $\mathcal{C}_i$, the screening score is the average

$$u_i = \frac{1}{|\mathcal{C}_i|} \sum_{C \in \mathcal{C}_i} u_i(C) = \frac{1}{|\mathcal{C}_i|} \sum_{C \in \mathcal{C}_i} \left\langle g_i^C, \bar{e}_i^C - \mathrm{Emb}(x_i^{(0)}) \right\rangle, \tag{16}$$

with g_i^C and $\bar{e}_i^C$ as in Eqs. 6–7. The $K_{\mathrm{short}} = 2B$ positions with the largest u_i form the shortlist.

Probe cost. Phase I-B evaluates the source reward R_0, the K_{short} single-site replacements, and the $\binom{K_{\mathrm{short}}}{2}$ pairwise replacements,

$$N_{\mathrm{probe}} = 1 + K_{\mathrm{short}} + \binom{K_{\mathrm{short}}}{2}, \tag{17}$$

all of which are batched. R_0 and the single-site values are reused to form $\hat{s}_i$ and $\hat{\Gamma}_{ij}$ in Eq. 8, whichever single-site term supplies s_i in Q. The pair term is a second finite difference of the learned reward under soft replacements: it measures departure from additivity under the reward model, not experimentally measured epistasis.

Pairwise expansion. With the singleton embeddings fixed, let $\mathcal{F}(S) = R_\omega(x^{(0)}[S \leftarrow \bar{e}_S])$ be the soft-replacement reward of §3.2 for $S \subseteq C_{\mathrm{short}}$, with $\mathcal{F}(\varnothing) = R_0$. Its inclusion–exclusion coefficients

$$d(W) = \sum_{A \subseteq W} (-1)^{|W|-|A|} \mathcal{F}(A) \qquad \text{satisfy} \qquad \mathcal{F}(S) = \sum_{W \subseteq S} d(W), \tag{18}$$

because the coefficient of $\mathcal{F}(A)$ in $\sum_{W \subseteq S} d(W)$ is $\sum_{A \subseteq W \subseteq S} (-1)^{|W|-|A|} = (1-1)^{|S \setminus A|}$, which is one for $A = S$ and zero otherwise. In particular $d(\varnothing) = R_0$, $d(\{i\}) = \hat{s}_i$, and $d(\{i,j\}) = \hat{\Gamma}_{ij}$, so with $s_i = \hat{s}_i$ and $\alpha = 1$, Eq. 9 gives

$$\mathcal{F}(S) - R_0 = Q(S) + \sum_{\substack{W \subseteq S \\ |W| \geq 3}} d(W). \tag{19}$$

Q therefore recovers the soft reward change exactly for sets of at most two positions and keeps the first- and second-order terms for larger sets, with higher-order interactions as the residual. The identity does not bound the gap to the best discrete substitutions or to $V_\beta(S)$, and the T4 choice $s_i = u_i$ changes the singleton terms, so it does not satisfy Eq. 19.

Subset solver. The solver enumerates the shortlisted subsets when this is tractable ($\binom{16}{8} = 12{,}870$ subsets at $B = 8$) and otherwise uses greedy forward selection followed by single-site swaps ($B = 16$). The score Q accepts additional pairwise terms, such as a StructureDCA family-plausibility penalty; no such term is active in any reported experiment.

C.4 Phase II details

Update direction. On the open anchors F, the optimized distribution factorizes, $q_\psi(a_F) = \prod_{i \in F} q_i(a_i)$ with $q_i = \mathrm{softmax}(\psi_i/\tau)$, while the reward couples the positions because it is evaluated

on the full sequence. Differentiating the sitewise softmax gives

$$\begin{aligned}\frac{\partial q_i(b)}{\partial \psi_i(a)} &= \frac{q_i(b)}{\tau}\big(\mathbf{1}\{a=b\} - q_i(a)\big),\\ \frac{\partial \bar{e}_i}{\partial \psi_i(a)} &= \sum_b \frac{\partial q_i(b)}{\partial \psi_i(a)}\mathrm{Emb}(b) = \frac{q_i(a)}{\tau}\big(\mathrm{Emb}(a) - \bar{e}_i\big).\end{aligned} \tag{20}$$

The straight-through construction in Eq. 5 passes this Jacobian unchanged, since the stop-gradient term has zero derivative. Taking its inner product with the reward gradient $g_i = \nabla_{e_i} R_\omega$, evaluated at the hard forward input, gives

$$\left[\widehat{\nabla}_{\psi_i} \widetilde{R}_F\right]_a = \frac{q_i(a)}{\tau}\left\langle g_i, \mathrm{Emb}(a) - \bar{e}_i \right\rangle, \tag{21}$$

which is Eq. 12. Because g_i is evaluated at the hard input while the Jacobian is taken through the soft expectation, the update approximates rather than equals the gradient of the discrete objective.

Decoding schedules. *Parallel masked commitment (ESM-2).* The thermostability and solubility experiments apply one joint update of 32 SGD steps to the full anchor set, exclude each source residue, and decode all anchors in parallel by argmax; the T4 artifacts use the same rule with the settings in App. A.4. *Iterative masked commitment (DPLM).* One position is decoded per round: the most confident open position is committed to its non-source residue, the remaining anchors are remasked, and the base model is queried again, so later substitutions condition on earlier ones. *Inverse-folding schedule (LigandMPNN, SolubleMPNN).* A fixed autoregressive order is chosen; the model predicts, tilts, and commits one substitution at a time, updating the sequence context while the backbone stays fixed, without reranking the remaining positions.

C.5 Iterative refinement

After a pass, each anchor $i \in S$ is ranked by its cached contribution to Q,

$$c_i(S) = \hat{s}_i + \alpha \sum_{j \in S \setminus \{i\}} \hat{\Gamma}_{ij}, \tag{22}$$

which equals the marginal $Q(S) - Q(S \setminus \{i\})$ when $s_i = \hat{s}_i$. A refinement round keeps the strongest edits, restores the reconsidered positions to the source residues, selects replacement anchors, and decodes them in the context of the fixed edits; a schedule sets how many positions are reconsidered in each round. Caching the Phase I scores avoids repeating every probe, but refinement remains a search heuristic: substitutions and base-model probabilities change with the sequence context, so the mutation budget is preserved while reward improvement is not guaranteed.

D Extended related work

Protein generative models. Protein models provide different priors over sequence and structure. ESM-2 and ESM-3 [Lin et al., 2023, Hayes et al., 2025] predict masked residues from sequence context; DPLM and EvoDiff [Wang et al., 2024, Alamdari et al., 2023] generate sequences through discrete diffusion; and ProGen2 and ProtGPT2 [Madani et al., 2023, Ferruz et al., 2022] use autoregressive factorization. Structure-conditioned inverse-folding models, including ProteinMPNN, LigandMPNN, SolubleMPNN, and ESM-IF1 [Dauparas et al., 2022, 2025, Goverde et al., 2024, Hsu et al., 2022], predict residues compatible with a supplied backbone. Another approach optimizes sequences directly through differentiable structure predictors [Norn et al., 2021]. APEX uses the per-position amino-acid probabilities from each family while preserving how each model predicts and decodes residues.

Additional optimization methods. Beyond the baselines discussed in §2, AdaLEAD [Sinai et al., 2020] alternates surrogate fitting with local exploration, and Deep Exploration Networks [Linder et al., 2019, 2020] train a dedicated generator against surrogate and diversity objectives. LaMBO-2 [Gruver et al., 2023] combines Bayesian optimization, guided discrete diffusion, and a task-specific surrogate. A further family of model-based optimization methods, including DynaPPO [Angermueller et al.,

2020], GFN-AL [Jain et al., 2022], DbAS and CbAS [Brookes and Listgarten, 2018, Brookes et al., 2019], and Feedback-GAN [Gupta and Zou, 2019], trains a policy, generator, or proposal distribution against the oracle across acquisition rounds and measures progress in oracle queries. APEX addresses a different setting: it emits a design for a given source in one shot, without training a generator, and is compared at matched budget on per-design quality.

Machine-learning-assisted directed evolution. Directed evolution and deep mutational scanning produce the variant–fitness measurements that support modern protein predictors [Arnold, 2018, Romero and Arnold, 2009, Fowler and Fields, 2014, Notin et al., 2023b,a, Meier et al., 2021]. Stability-focused examples include Mutate Everything [Ouyang-Zhang et al., 2023] and Stability Oracle [Diaz et al., 2024]; evolution-inspired representation learning provides another route to more informative predictors [Gong et al., 2024]. Machine-learning-assisted directed evolution aims to prioritize useful variants from limited assay data [Wittmann et al., 2021, Biswas et al., 2021], including recent work on multi-residue interactions [Tran et al., 2026, Visani et al., 2026]. APEX operates at inference time once a differentiable task predictor is available, avoiding retraining of the generative base model for each property.

Reward overoptimization. Optimizing a learned reward can favor sequences that score well in silico without improving the underlying property [Gao et al., 2023, Skalse et al., 2022]. Protein-design studies therefore commonly pair the target score with sequence-, evolutionary-, and structure-based checks [Huang et al., 2025, Kirjner et al., 2024, Emami et al., 2023]. APEX initializes residue logits from a frozen protein prior and limits the number of edits, which supplies a useful inductive bias but does not rule out reward-model exploitation. This motivates the complementary proxy metrics and experimental-validation caveats used in our evaluation.

E Limitations and broader impacts

E.1 Limitations

APEX inherits the strengths and failure modes of its reward model. The thermostability, solubility, and T4 gains are predictions, while PLL, BLOSUM62, StructureDCA, pLDDT, and backbone similarity are complementary proxies rather than experimental measurements. GB1 provides measured DMS values, but its reward model is trained on the same dataset, so this is not an independent generalization test. Prospective assays are needed to establish improved stability, solubility, or function.

The exhaustive Phase I subset search scales as $\binom{K_{\text{short}}}{B}$ and is therefore used only when the shortlist problem is small; the $B = 16$ experiments use greedy selection followed by single-site swaps (App. C.3). The model families also predict and decode residues differently, and optional refinement is a search heuristic rather than a monotonic-improvement guarantee.

E.2 Broader impacts

Fixed-budget computational design may reduce experimental screening by prioritizing a smaller set of candidates, and the frozen base model can be reused once an appropriate differentiable property model is available. The same flexibility also transfers the limitations and potential misuse of that property model to the design procedure. The evaluators studied here target thermostability, solubility, and GB1 binding rather than pathogenicity-related properties, but applications to other objectives require their own biosafety review and experimental controls.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and §1 describe an inference-time, fixed-budget method with frozen base and reward models. The method screens candidate sites, includes pairwise reward interactions in anchor selection, and decodes substitutions with family-specific decoding schedules. Empirical claims distinguish learned target-property predictions from plausibility proxies; the GB1 DMS lookup is identified as non-independent, and T4 is presented as a computational case study.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: App. E.1 discusses reward-model dependence, proxy-based evaluation, Phase I scaling, family-specific decoding schedules, and the heuristic nature of refinement. It also explains why the GB1 DMS lookup is not an independent generalization test and notes the need for prospective assays.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Apps. C.1, C.3, and C.4 state the assumptions and derive the KL and Gibbs identities, the fixed-proposal pairwise expansion, and the STE update direction. These derivations do not establish global optimality or monotone improvement for the implemented heuristics.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [No]

   Justification: App. A documents the datasets, evaluators, metrics, baseline conventions, and primary APEX settings. Exact end-to-end reproduction is not currently possible because the code release (`https://github.com/skyzczzz/APEX`) is pending and trained reward checkpoints are not provided with this paper, and the archived GB1 and T4 runs do not preserve all generation settings and cohort mappings (Apps. A.1 and A.4).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code will be released at https://github.com/skyzczzz/APEX; run artifacts and trained reward checkpoints are not provided with this paper. The paper provides algorithmic definitions, pseudocode, hyperparameters, and per-method protocols, while the source datasets and base models are cited.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [No]

   Justification: Apps. A.1 and A.3 define the primary settings, metrics, and baseline conventions, but the GB1 generation settings and the T4 design-set mappings could not be fully recovered from the archived runs (Apps. A.1 and A.4), so the answer is No.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper reports descriptive means over $N{=}100$ designs per cell and does not provide source-paired hypothesis tests or repeated-run uncertainty for any method. No statistical-significance claim is made.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: App. A.1 reports the accelerator, representative THERM/SOL and GB1 runtimes, and total project compute, but not a per-run resource breakdown for every reported method and setting.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The work produces in silico protein-design recommendations for thermostability, solubility, and GB1 binding using cited research datasets. It involves no human subjects or private personal data and does not release a new generative model.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: App. E.2 discusses candidate prioritization, the need for a property-specific differentiable evaluator, inherited model risks, and application-specific biosafety review.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new pretrained generative protein model or scraped dataset. The task evaluators considered here do not target pathogenicity or virulence.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: Existing assets are cited, and App. A.5 records the available licenses. Exact historical revisions and some dataset and checkpoint terms remain unverified.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new datasets or model checkpoints are released with this paper; the pending code release will carry its own documentation. The method and task-specific evaluators are described in the manuscript.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The reported evaluations involve no crowdsourcing or human participants.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The reported computational evaluations involve no human participants; this approval item is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: APEX uses pretrained protein language models as frozen sequence generators and differentiable property evaluators. Their roles, architectures, and inference procedures are described in §3, §4, and App. A.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.